Universidad
Carlos III de Madrid

INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN
TELEMÁTICA

PROYECTO FIN DE CARRERA

Desarrollo de una aplicación NFC en un entorno universitario con autenticación basada en el Elemento Seguro.

Autora: Beatriz Juárez Gutiérrez
Tutor: Mario Muñoz Organero

Leganés, Enero de 2011

Si buscas resultados distintos,
no hagas siempre lo mismo.

Albert Einstein

Agradecimientos

A Sara, por todas las horas de clases, de prácticas y de biblioteca, que han sido unas cuantas, por esos mixtos de la cafetería, esos menús del Oh La Lá y esas escapaditas a parquesur y a teleppiza…Por hacerlas mucho más amenas y por estar siempre ahí durante toda la carrera, porque no sé que habría hecho sin ti...

A mi familia y mis amigos, por apoyarme y escucharme siempre, aunque no entendiérais nada de lo que os decía (a veces yo tampoco).

A mi novio Viti, por su paciencia infinita y la manera en la que siempre consigue sacarme una sonrisa.

A Mario, sin el que la realización de este proyecto habría sido imposible, por su dedicación y preocupación durante todos estos meses, y por abrirme las puertas para poder conocer la tecnología NFC.

A Gamma Solutions, que me ha dado la oportunidad de realizar el proyecto y de conocer el ambiente de trabajo en la empresa real. Me he sentido muy agusto durante todo el tiempo que he pasado allí.

Resumen

Hoy en día prácticamente todo el mundo posee un teléfono móvil, son aparatos electrónicos generalmente muy sencillos de utilizar y muy portables (pesan poco, caben en un bolsillo…) lo que hace muy atractivo el desarrollo de aplicaciones móviles.

NFC es una tecnología que nos ofreces grandes ventajas como pagar con el móvil sin necesidad de tener llevar dinero encima, simplemente pasando el móvil por el lector. También permite el control de acceso sin necesidad de que haya un guardia de seguridad en la entrada y compruebe si nuestra tarjeta de acceso es válida o no.

El elemento seguro que posee el móvil proporciona identificación, pago o almacenamiento de información de manera segura.

Otra tecnología muy importante es Bluetooth, que permite un intercambio de datos de forma intuitiva y también segura ya que requiere un PIN.

La aplicación desarrollada para el proyecto posee todas estas ventajas: se trata de un móvil con la tecnología NFC que permite la identificación de forma segura y fiable a través del elemento seguro del móvil. Esta aplicación permite la autenticación en distintas ubicaciones de la universidad (despachos, laboratorios, biblioteca...) teniendo almacenado en el elemento seguro el NIA y una clave que se devolverá cifrada para aumentar, todavía más, la seguridad de la aplicación. La aplicación permite recibir información de un lector a través de NFC, el móvil muestra las posibles opciones, se elege la deseada y se envía al lector. Si la opción deseada no requiere autenticación, el lector envía la información al servidor y, si la requiere la solicita. En este último caso el móvil debe acceder al elemento seguro para obtener la contraseña, el elemento seguro la devuelve cifrada, el móvil se la envía al lector y éste al servidor, que comprueba si es válida. Si es correcta se envía la información al móvil, y sino, le indica que la contraseña es invalida.

La aplicación también utiliza Bluetooth para enviar a un lector la reserva realizada, con el código y la información de la misma, y de este modo obtener de manera rápida y sin confusiones la tutoría, el libro o el menú reservados. También se puede enviar mediante NFCIP, dependiendo de las tecnologías soportadas por el lector que recibe la reserva.

El proyecto tiene una segunda parte, que consiste en una aplicación web cuya función es similar a la del móvil pero sin necesidad de acudir a las distintas ubicaciones de la universidad. Para acceder a la aplicación será necesario un usuario y un password y, en función de si es un alumno o un profesor, se mostrarán unas opciones u otras.

Abstract

Nowadays almost everyone owns a mobile phone, electronic devices are generally very simple to use and very portable (low weight, fit in a pocket ...) which makes mobile application development very attractive.
NFC is a technology that offers great benefits and pay with your mobile phone without having to carry money, simply swiping a phone over the reader. It also allows control of access without need for a security guard at the entrance to check if our access card is valid or not.
The element that owns the mobile insurance provides identification, credit or store information securely.
Another important technology is Bluetooth, which allows exchange of data in an intuitive and secure because it requires a PIN.

The application developed for the project has all these advantages: it is a phone with NFC technology that allows the identification of safe and reliable through the secure element of the mobile. This application provides authentication in various university locations (offices, laboratories, library ...) . NIA and a key are stored in the secure element and are returned encrypted to increase even more, the security of the application. The application allows a reader to receive information via NFC, the mobile shows the possible options, choose the desired and sent to the reader. If the desired option does not require authentication, the reader sends the information to the server and, if required upon request. In the latter case, the mobile must access the secure element to obtain the password, the secure element returns the information encrypted, the mobile sends it to the reader who sends it to the server, who checks if is valid. If so, the information is sent to your phone, and if not, tells the mobile that the password is not valid.
The application also uses Bluetooth to send a reader the booking with the code and the information itself, in order to obtain quickly and without confusion the tutorial, the book or the menu reserved. Bookings can also be sent by NFCIP, depending on the technologies supported by the reader who receives the information.

The project has a second part, which consists of a web application whose functions are similar to mobile phone without having to go to different locations of the university. The application access will require a user name and password and, depending on whether a student or a teacher, will show some other options.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y objetivo

1.1 Introducción

Hasta hace poco tiempo los teléfonos móviles se utilizaban únicamente para realizar llamadas y enviar mensajes de texto. En la actualidad, los teléfonos móviles han evolucionado y han mejorado sus características y ofrecen muchísimas más ventajas: tienen conexión a Internet, se puede escuchar la radio, realizar videollamadas, instalar infinidad apliaciones…

El uso de las tarjetas también ha aumentado de forma significativa durante estos últimos años, se utilizan tarjetas de crédito para pagar, tarjetas que nos proporcionan distintos establecimientos como restaurantes o gasolineras para acumular puntos y luego tener derecho a descuentos o regalos. Actualmente ha surgido el DNI electrónico que lo entregan junto con un PIN que nos permite autenticarnos y realizar pagos por internet de manera segura y controlada.

La tecnología NFC es una tecnología muy reciente pero que ofrece muchas posibilidades. Cuenta con muchas ventajas: es muy fácil de utilizar y muy intuitiva, es bastante segura y eficiente, lo que la hace idónea para intercambios de información entre dos terminales.

La tecnología Bluetooth, que está bastante mas desarrollada y estudiada que NFC es también muy útil para realizar intercambios de información y además cuenta con la ventaja de que en la actualidad la mayor parte de los teléfonos móviles ya traen esta tecnología integrada.

Al poner en común todos estos elementos y tecnologías, obtenemos un montón de ventajas y surge la idea de desarrollar una aplicación de control de acceso. Esta aplicación llevará a cabo la autenticación de los usuarios en distintos lugares de la universidad para poder reservar un ordenador, solicitar una tutoría, reservar un libro, consultar las características de los ordenadores de un laboratorio, conocer el horario de un aula… La aplicación se encuentra dentro de un teléfono móvil que posee las tecnologías NFC y Bluetooth y además que se puede acceder a su elemento seguro para obtener el usuario y la contraseña cifrada que permitirá la autenticación con el terminal NFC que se encuentre en las distintas ubicaciones de la universidad pudiendo así tener acceso a la información y a realizar reservas. Posteriormente estas reservas se podrán enviar a un lector especializado mediante NFCIP o Bluetooth, según se desee o según sea soportado por el lector y posteriormente se tendrá acceso al recurso reservado.

Además, todas estas tareas que se pueden realizar a través del móvil, se pueden realizar a través de una aplicación web que, tras la autenticación con usuario y contraseña, permite reservar tutorías con profesores, reservas libros o incluso un menú de la cafetería.

Esta aplicación se ha desarrollado para facilitar el acceso a la información y poder realizar reservas tanto para los alumnos como para los profesores. Simplemente con llevar un móvil encima, como hacemos habitualmente, podemos solicitar una tutoría o reservar un menú y los profesores pueden, con simplemente tocar el lector, consultar qué alumno y para cuándo ha reservado una tutoría de las que el profesor tenía libres. La segunda parte del proyecto consiste en una aplicación web que permite realizar las mismas opciones que con el móvil pero desde un ordenador. También hay algunasopciones añadidas: los profesores pueden no sólo consultar las tutorías fijadas, sino añadir tutorías libres o eliminar alguna tutoría que todavía no hubiera sido solicitada. También pueden consultar una gráfica con los recursos más accedidos por los alumnos.

Esta aplicación prentende agilizar y facilitar las operaciones cotidianas que se llevan a cabo en la universidad. Por ejemplo, un profesor puede añadir una tabla con las tutorías que tiene disponibles, indicando día y hora y el alumno, en lugar de solicitarle la tutoría por correo electrónico y esperar a que el profesor le responda si sigue disponible o no, simplemente con el móvil el lector le manda las tutorías disponibles, elige la que desea, se lo comunica al lector y al siguiente alumno que solicite una tutoría esa ya no le aparece disponible. El profesor con tocar con su móvil el lector, sabe qué tutorías y qué alumnos las han solicitado, sin necesidad de responder por correo si la tutoría esta disponible o no, pues ya está fijada.

1.2 Objetivos

El objetivo fundamental de la tesis es el de proporcionar de una forma rápida y segura el acceso a la información de distintos recursos de la universidad y que sea accesible para todos con tan sólo llevar un móvil o tener un ordenador disponible.

Con esta aplicación se quiere demostrar cómo mediante la tecnología NFC se proporciona una solución muy eficaz para el acceso a la información de forma rápida y más segura que otras tecnologías inalámbricas, ya que las transacciones NFC sólo se pueden activar en un rango de acción muy limitado. Como alternativa a NFC surge Bluetooth, pues al ser una tecnología más desorrallada se encuentra en mayor cantidad de dipositivos que NFC, y la información se envía de manera segura al requerir un PIN en ambos extremos y los datos se mandan encriptados. Es el usuario el que elige a través de cual de estas tecnologías envía su reserva al lector. Para añadir mayor seguridad a la aplicación, la autenticación se realiza a través del elemento seguro y, además, de una manera cifrada. Con todo esto conseguimos que la aplicación sea rápida, segura y con alternativas en caso de que NFC no este soportado (mediante Bluetooth) o que no se disponga de móvil con la tecnología NFC (mediante la aplicación web complementaria).

En base a ese objetivo principal, se proponen los siguientes objetivos parciales:

- Existen dos tipos de usuarios: profesores y alumnos.
- La autenticación se realizará mediante elemento seguro del móvil o mediante usuario y password en caso de la aplicación web.
- Los profesores podrán crear y eliminar tutorías libres y consultar tutorías fijadas.
- Los alumnos podrán consultar las tutorías libres de los profesores y solicitarlas.
- Todos los usuarios podrán consultar información de las distintas ubicaciones (horarios, características de los ordenadores, disponibildad de laboratorios…).
- Todos los usuarios podrán reservar libros, tutorías, ordenadores o menús de la cafetería siempre y cuando estén debidamente autenticados.
- Las reservas realizadas se almacenarán en la base de datos para posteriormente poder ser consultadas y enviadas al lector correspondiente mediante NFCIP y Bluetooth.
- Todos los usuarios podrán acceder a la aplicación web y, tras la autenticación, podrán realizar las mismas acciones que con el teléfono móvil.

1.3 Fases del desarrollo

El plan de trabajo es el representado en la figura 1.1 mediante un diagrama de Gantt. A continuación se detalla cada una de éstas fases.

- **Fase 1: Estudio de la tecnología NFC.** En esta fase se realiza un estudio de la tecnología y se experimenta con ella realizando diversas aplicaciones.
- **Fase 2: Elección de la aplicación**. Búsqueda de un proyecto en el que se desarrolle una aplicación útil, que se pueda emplear de forma viable y que se contruya a partir de tecnologías emergentes.
- **Fase 3**. **Características de la aplicación.** Una vez escogida la finalidad de la aplicación, se establecen las pautas que debe cumplir: a quén va dirigida, que funcionalidades debe ofrecer, como se lleva a cabo la autenticaión…
- **Fase 4**. **Estudio de alternativas.** Tras elegir la tecnología NFC como principal se propone Bluetooth como alternativa para los casos en los que la tecnología NFC no esté soportada en alguno de los dos extremos o la comunicación falle.
- **Fase 5. Implementación de la aplicación.** Esta fase comprende todo el desarrollo de la aplicación móvil que se puede dividir en tres subfases:
    - **Fase 5.1. Acceso al elemento seguro.** En esta fase se estudió y desarrolló el acceso al elemento seguro del móvil.
    - **Fase 5.2. Implementación NFC.** En esta fase se estudió y desarrollo la tecnología NFC para que cumpliera los objetivos de la aplicación.
    - **Fase 5.3. Implementación Bluetooth. .** En esta fase se estudió y desarrollo la tecnología Bluetooth para que sirviera como alternativa.

- **Fase 6. Integración de la aplicación.** Esta fase comprende la integración de todas las tecnologías desarrolladas en la fase previa.
- **Fase 7**. **Creacción de la aplicación web complementaria.** Tras el desarrollo de la aplicación con la tecnología NFC y su alternativa, Bluetooth, con autenticación mediante el elemento seguro, se propone la creación de una aplicación web complementaria que ofrezca las mismas características que la aplicación móvil para aquellas personas que no dispongan de móvil von NFC o que les sea más como utilizar un ordenador.
- **Fase 8. Pruebas.** Una vez realizadas tanto la aplicación móvil como la aplicación web, se realiza una evaluación del sistema completo para comprobar que se cumplen todos los requisistos establecidos en la Fase 2.
- **Fase 9. Documentación.** Elaboración de una memoria en la que se explica con detalle la finalidad de la aplicación, se describen las tecnologías empleadas y los requisitos necesarios para llevarla a cabo.

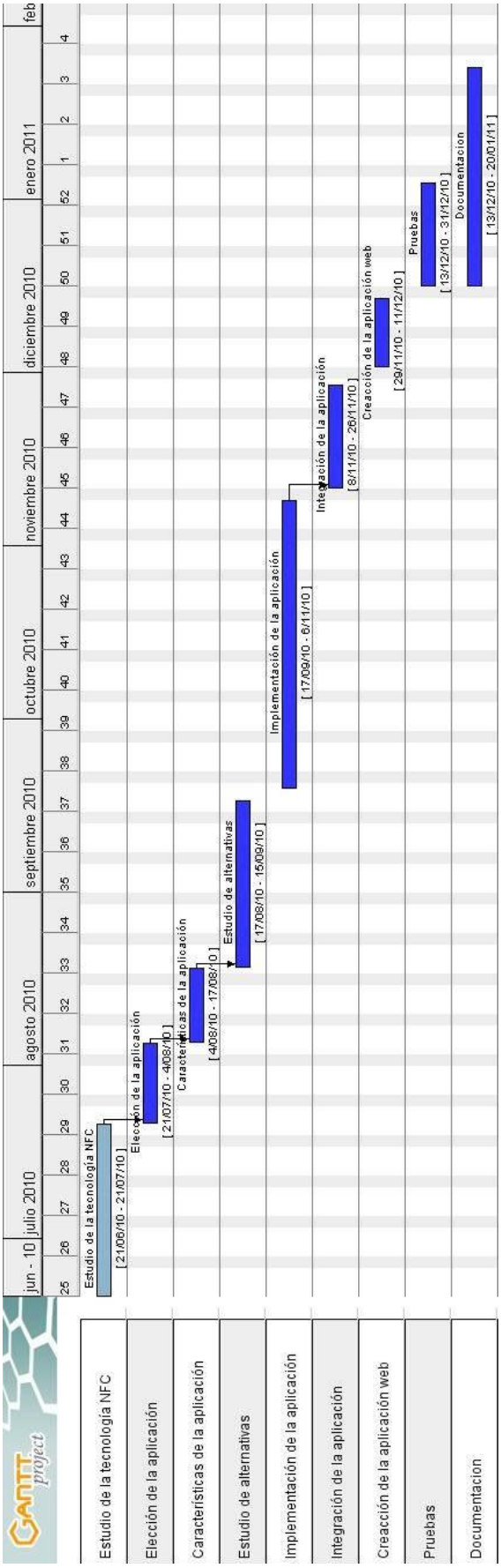


Figura 1.1 Diagrama de Gantt

1.4 Estructura de la memoria

Para facilitar la lectura de la memoria, se incluye a continuación un breve resumen de cada capítulo:

- **Capítulo 2-Estado del Arte:** Estudio detallado de las tecnologías utilizadas para la realización del proyecto.
- **Capítulo 3- Descripción del escenario:** Descripción de las posibles situaciones que pueden tener lugar en cada aplicación, tanto la móvil como la web. En este capítulo también se describen los requisitos software y hardware necesaros para ejecutarlas.
- **Capítulo 4- Descripción de la Aplicación:** Exposición de los distintos proyectos y clases desarrolladas para la ejecución de la aplicación. Desarrollo de un diagrama UML que facilite la compresión de la relación entre las mismas.
- **Capítulo 5- Casos de uso:** En este capítulo se muestran las posibles ubicaciones en las que se puede desarrollar el proyecto, así como las pantallas que irían apareciendo a lo largo de la ejecución. Se muestran las posibles opciones de cada ubicación y si requiere autenticación o, por el contrario, es información pública.
- **Capítulo 6-Pruebas:** Descripción de las pruebas realizadas para comprobar la correcta funcionalidad de la aplicación.
- **Capítulo 7-Conclusiones y trabajos futuros:** Exposición sobre las conclusiones generales obtenidas tras la implementación del proyecto: NFC, Bluetooth y JavaCard y posibles trabajos futuros propuesto para contribuir a la mejora del proyecto desarrollado.
- **Capítulo 8-Presupuesto:** Cálculo del presupuesto necesario para el desarrollo de la aplicación, incluye tanto costes materiales como de personal.

Capítulo 2

Estado del Arte

2.1 Introducción

Para poder comprender el funcionamiento del proyecto es necesario realizar una breve descripción de las tecnologías que han sido necesarias para su desarrollo.

Se comenzará haciendo una descripción de NFC, la tecnología más importante que constituye la base del proyecto.

Posteriormente se descibirá la tecnología Bluetooth, que también constituye una parte importante del proyecto ya que sirve como alternativa a NFCIP en el envío de las reservas realizadas.

A lo largo de este capítulo también se realiza una breve explicación del lenguaje de programación Java, que ha sido el lenguaje utilizado para desarrollar el proyecto y se presenta también J2ME, que permite el desarrollo de aplicaciones móviles.

Otra de las tecnologías que intervienen en el proyecto es JavaCard, una tecnología muy necesaria para el desarrollo de aplicaciones en tarjetas inteligentes que además permiten la elaboración de aplicaciones más seguras.

El servidor web escogido para el desarrollo de la aplicación es Tomcat y la base de datos Derby. En este capítulo se explican sus características.

Para la aplicación web es necesario el desarrollo de JSP, y para ambas aplicaciones, tanto la web cómo la móvil, es necesario el uso de servlets ya que son los que gestionan las peticiones recibidas y se encargan de las consultas a la base de datos.

Todas estas tecnologías se explican detalladamente a continuación para facilitar la compresión del proyecto.

2.2 NFC

2.2.1 Descripción

Near Field Comunication, NFC, es una tecnología de comunicación inalámbrica de corto alcance que permite el intercambio bidireccional de datos entre dispositivos a una distancia corta inferior a 20 centrímetros.

Su desarrollo empieza en el año 2002, sus promotores fueron principalmente Philips y Sony que buscaban conseguir compatibilidad con sus tecnologías Mifare y FeliCa respectivamente. La idea de desarrollar esta tecnología fue crear un nuevo protocolo que tuviera compatibilidad con las tecnologías sin contacto de corto alcance ya existentes, razón por la que NFC es una extensión simple del estándar ISO/IEC 14443-5 de tarjetas de proximidad (tarjetas RFID sin contacto) que combina la interfaz de una tarjeta inteligente y de un lector dentro de un mismo dispositivo.

La tecnología NFC permite el intercambio de datos entre dispositivos a través de un diálogo. En este protocolo siempre hay uno que inicia la conversación y es este el que monitorizará la misma, este rol es intercambiable entre las dos partes implicadas. Existe una interacción bidireccional entre los dispositivos electrónicos que participan en la comunicación. Es una comunicación half-duplex (ambos sentidos, pero no simultáneamente), ya que se emplea una única portadora a 13,56 MHz, lo que implica que no se aplique ninguna restricción y no requiera ninguna licencia para su uso. Las velocidades de transmisión soportadas actualmente son de 106, 212 y 424 Kbps.

Un dispositivo NFC puede comunicarse con otros dispositivos NFC o con otras infraestructuras inalámbricas y sin contactos ya existentes. Principalmente, se utiliza en dispositivos móviles ya que en la actualidad, la mayor parte de la población posee su propio móvil.

Figura 2.1: Ejemplo de funcionamiento NFC

2.2.2 Evolución

NFC surge a partir de la tecnología RFID (siglas de Radio Frequency IDentification), de hecho la tecnología NFC se compone de integración de la tecnología RFID con otras tecnologías interconectadas. Al estar basado en RFID, es necesario un lector y una tarjeta.

2.2.2.1 RFID

En la actualidad, es la tecnología más extendida para la identificación de objetos es la de los códigos de barras. Sin embargo, éstos presentan algunas desventajas, como la escasa cantidad de datos que pueden almacenar y la imposibilidad de ser reprogramados. La mejora ideada constituyó el origen de la tecnología RFID; consistía en usar chips de silicio que pudieran transferir los datos que almacenaban al lector sin contacto físico, de forma equivalente a los lectores de infrarrojos utilizados para leer los códigos de barras.
El modo de funcionamiento de los sistemas RFID es simple. La etiqueta RFID, que contiene los datos de identificación del objeto al que se encuentra adherido, genera una señal de radiofrecuencia con dichos datos. Esta señal puede ser captada por un lector RFID, el cual se encarga de leer la información y pasarla en formato digital a la aplicación específica que utiliza RFID. [1]

Un sistema RFID consta de los siguientes tres componentes:

- **Etiqueta RFID**: compuesta por una antena, un transductor radio y un material encapsulado o chip. El propósito de la antena es permitirle al chip, el cual contiene la información, transmitir la información de identificación de la etiqueta. Existen varios tipos de etiquetas. El chip posee una memoria interna con una capacidad que depende del modelo y varía de una decena a millares de bytes. Existen varios tipos de memoria:

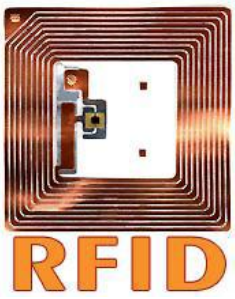


Figura 2.2: Etiqueta RFID

  - Solo lectura: el código de identificación que contiene es único.
  - De lectura y escritura: la información de identificación puede ser modificada por el lector.
  - Anticolisión: Se trata de etiquetas especiales que permiten que un lector identifique varias al mismo tiempo (habitualmente las etiquetas deben entrar una a una en la zona de cobertura del lector).

- **Lector de RFID o transceptor:** compuesto por una antena, un transceptor y un decodificador. El lector envía periódicamente señales y cuando capta una señal de una etiqueta (que contiene la información de identificación), extrae la información y la pasa al subsistema de procesamiento de datos.

Figura 2.3: Lector RFID

- **Subsistema de procesamiento de datos o Middleware RFID:** proporciona los medios de proceso y almacenamiento de datos.

2.2.2.2 De RFID a NFC

Near Field Communication, es una tecnología de radiofrecuencia de corto alcance que permite la comunicación entre diferentes dispositivos hasta 10 centímetros. Aunque supone un avance dentro del campo de la investigación de RFID. La principal diferencia con RFID es que NFC permite que los dos dispositivos establezcan una comunicación casi de igual a igual, lo que facilita el intercambio de datos entre los dos. Además, tiene un radio de acción más pequeño, lo que proporciona mayor privacidad y seguridad. Otra diferencia es la rapidez y la sencillez en el intercambio de datos. Ya no es necesario navegar por complicados menús para emparejar dos dispositivos. Con un simple toque el dato que está en el dispositivo móvil NFC se transfiere a otro dispositivo automáticamente. [2]

2.2.3 Comparación con otras tecnologías

Es preciso aclarar que no está dirigida a la transmisión masiva de datos, al estilo de tecnologías como WLan o Bluetooth, sino a la comunicación entre dispositivos con capacidad de proceso como teléfonos móviles, PDA o PCs, entre sí, o como lectores de etiquetas.[3]

	NFC	RFID	IrDa	Bluetooth
Tiempo de establecimiento	<0,1ms	<0,1ms	~0,5 s	~6 s
Alcance	10cm	3m	5m	30m
Facilidad de uso	Fácil, intuitivo y rápido	Fácil	Fácil	Media
Seguridad	Muy alta	Buena	Visión directa	Mediante PIN
Usos	Pago, acceso, compartir información, etc.…	Identificación, seguimiento, etc.…	Control e intercambio de datos	Red para intercambio de datos

Tabla 2.1: Comparativa entre distintas tecnologías.

2.2.4 Modos de funcionamiento

La comunicación entre dispositivos NFC se realiza a través de un diálogo entre un dispositivo denominado “Iniciador” y uno o varios dispositivos denominados “Destino”, debiendo responder éstos antes de recibir otra petición. Existen dos modos de establecer la comunicación: **activo y pasivo**. [4]

- **Modo pasivo**: el dispositivo Iniciador genera el campo electromagnético y el dispositivo destino se comunica con éste modulando la señal recibida. En este modo, el dispositivo destino obtiene la energía necesaria para funcionar del campo electromagnético generado por el Iniciador.

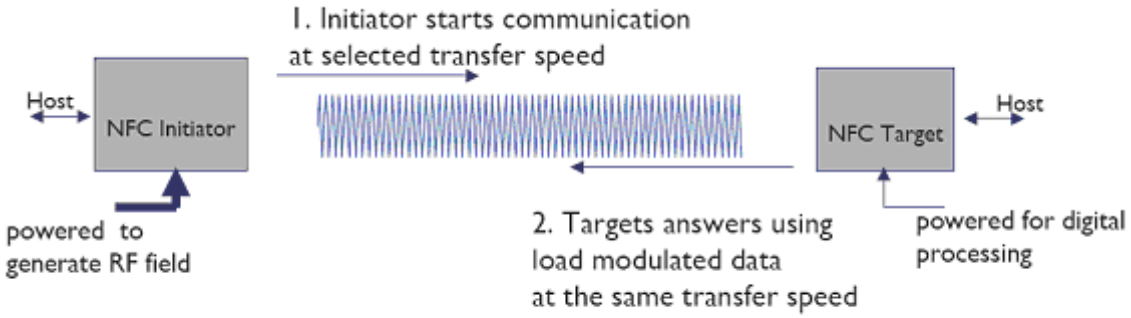


Figura 2.4: Esquema del modo de funcionamiento pasivo

- **Modo activo:** tanto el dispositivo Iniciador como el destino se comunican generando su propio campo electromagnético. En este modo, ambos dispositivos requieren de una fuente de alimentación para funcionar.

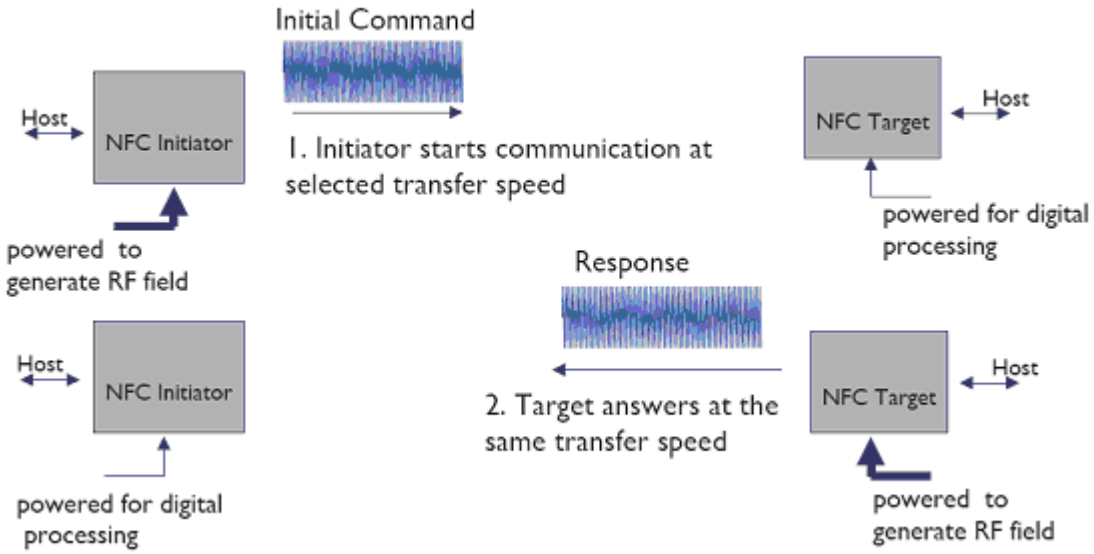


Figura 2.5: Esquema del modo de funcionamiento activo

Cuando el dispositivo funciona en modo pasivo, el receptor sólo se utiliza para establecer la comunicación y confirmar la recepción de los datos. Sin embargo, en modo activo, se requiere que ambos nodos negocien el intercambio de datos. Aunque muchas aplicaciones requieren que los dispositivos involucrados sean activos, la combinación de uso activo/pasivo puede ser útil para comunicarse con elementos sin batería, como las tarjetas sin contactos o las etiquetas RFID que no dispongan de fuente de alimentación propia.

2.2.5 Transacción NFC

Una transacción NFC siempre sigue una misma secuencia de operación que consta de los siguientes pasos:

- Descubrimiento de dispositivos NFC
- Autenticación
- Negociación
- Transferencia de información
- Confirmación

A bajo nivel, el protocolo NFC incluye un procedimiento para la autenticación segura y mecanismos anti-colisión para evitar la escucha del canal de comunicación. Para aplicaciones donde la seguridad es muy importante, como es el caso de su uso como medio de pago, es posible utilizar cifrado AES y triple DES, con lo que la seguridad se equipara a la ofrecida por las tarjetas inteligentes bancarias.

Durante la fase de negociación del protocolo NFC, se establecen parámetros que definen las características de la comunicación como son la velocidad de transmisión, el identificador del dispositivo, tipo de aplicación, tamaño de la transferencia y la acción solicitada.

Dado que NFC no está diseñado para una transferencia masiva de datos, se puede usar para configurar otras conexiones inalámbricas que ofrezcan mayor ancho de banda como son Bluetooth o Wi-Fi. Cabe destacar que un emparejamiento normal de dispositivos Bluetooth toma entre cinco y seis segundos, frente a los 200 ms que tarda la tecnología NFC.

2.2.6 Arquitectura dispositivo móvil NFC

Los principales componentes de un dispositivo móvil NFC son una bobina o antena incorporada en el interior del teléfono, el chip NFC y el denominado elemento seguro que es un chip con características de seguridad similares a las encontradas en las tarjetas inteligentes y que se encarga de procesar de forma segura las transacciones. El elemento seguro es uno de los componentes más importantes en la arquitectura hardware del dispositivo NFC. [5]

Actualmente, existen tres opciones para la ubicación del elemento seguro dentro del teléfono móvil NFC:

- Elemento seguro incorporado en la electrónica del móvil: el elemento seguro puede ser un microcontrolador incorporado en el dispositivo móvil, bien montado en la placa base directamente o conectado de alguna forma a la placa base. La principal ventaja es que el elemento seguro que puede ser incorporado tiene todas las certificaciones necesarias hardware y software demandadas por el sector bancario. Sin embargo, la principal desventaja de esta solución es su incapacidad de gestionar las credenciales de pago del usuario cuando éste quiere cambiar de teléfono móvil. Una posible solución a este problema es la gestión mediante el protocolo OTA (Over-The-Air) de igual forma en la que se produce la inicialización del elemento seguro.

- Tarjeta de memoria como elemento seguro: con esta solución, una tarjeta de memoria incorpora un chip seguro con un microcontrolador y memoria flash. Esta solución permite que terceras partes puedan suministrar tarjetas precargadas con su aplicación.

- Tarjeta SIM como elemento seguro: la tarjeta SIM incorpora la aplicación de pago, así como cualquier otra aplicación NFC. Respecto al software del dispositivo móvil, consiste en una serie de aplicaciones denominadas MIDlets que son descargadas en la memoria del teléfono NFC mediante el protocolo OTA (Over-The-Air) y engloban servicios para aplicaciones de crédito, programas de fidelidad del cliente, etc…, y que interactúan a su vez con otra aplicación denominada applet [ver sección 2.5.4] residente en el elemento seguro y que actúa como monedero de acuerdo a los estándares de pago PayPass de MasterCard oVisa Wave de Visa.

2.2.7 Etiquetas NFC

Las etiquetas, también denominadas *tags* o *transpondedores* son dispositivos que constan de tres elementos: una antena, un circuito integrado y un elemento almacenador de energía. La antena permite realizar la comunicación entre la etiqueta y el lector, y se debe tener en cuenta que su tamaño limitará la distancia máxima de lectura.

Figura 2.6: Distintos tipos de etiquetas.

En los ultimos años la producción de etiquetas ha tenido un decrecimiento 27 en el precio por unidad, aunque el coste se encuentra todavía por encima de los códigos de barras. Además existen otros aspectos como es la forma en la que las etiquetas se alimentan:

- **Etiquetas activas:** Una etiqueta activa necesita de una pequeña batería que le proporcione alimentación para poder generar y transmitir continuamente la señal de radiofrecuencia donde van codificados y modulados los datos. Pueden ser leídas por lectores que se encuentren a grandes distancias, llegando incluso a los 30 metros, y su capacidad de memoria le permite almacenar Kilobytes de información.

  Entre los inconvenientes nos encontramos con las interferencias que se producen con móviles y otros aparatos, su precio que es elvedado al requerir la batería y la vida últil de la batería que depende de muchos factores, lo que hace muy difícil prever cuando ocurrirá un fallo. Otro inconveniente es que su batería en ocasiones se descarga y como consecuencia se produce una pérdida total de la señal.

- **Etiquetas pasivas:** No contienen batería, utiliza campos electromagnéticos creados por los lectores que tienen un doble propósito, ya que a la vez obtienen información de ellas. La distancia de lectura de una etiqueta pasiva puede llegar hasta los 5 y 7 metros. Al trabajar con pequeños niveles de energía, tienen una capacidad de memoria relativamente baja.

  Presentan una gran ventaja sobre las etiquetas activas en cuanto a precio por unidad, por lo que se usan en más aplicaciones. De una manera gradual, a medida que el precio disminuye, entran en competencia con los códigos de barras.

  Existen 4 tipos básicos de etiquetas:

  - **NFC Tag Tipo 1:** estas etiquetas están basada en las especificaciones ISO-14443A, con capacidad de lectura y escritura. También es posible configurar el tag para que sea de sólo lectura. La capacidad de memoria es de 96 bytes extensible a 2 Kbytes. La velocidad de comunicación es de106 Kbits/s.

  - **NFC Tag Tipo 2:** basadas en las especificacionesISO-14443A, con capacidad de lectura y escritura o sólo lectura. La capacidad de memoria es de 48bytes, ampliable hasta 2 Kbytes. La velocidad de lectura es de sólo 106 Kbits/s.

  - **NFC Tag Tipo 3:** basadas en el estándar FeliCa, son de lectura y escritura o sólo de lectura. La capacidad de memoria es variable siendo su límite de 1Mbyte. La velocidad de comunicación es de 212 Kbits/s o424 Kbits/s.

  - **NFC Tag Tipo 4:** compatibles con el estándar ISO-14443A y B. Son configurados como de lectura y escritura o sólo de lectura. La capacidad de memoria es variable de hasta 32 KBytes con una velocidad de comunicación de 424 Kbits/s

	Type 1	Type 2	Type 3	Type 4
RF Interface	ISO 14443 A-2	ISO 14443 A-2	FeliCa (ISO 18092, passive communication mode at 212 kbits/sec)	ISO 14443-2
Initialization	ISO 14443 A-3	ISO 14443 A-3	FeliCa (ISO 18092, passive communication mode at 212 kbits/sec)	ISO 14443-3
Speed	106 kbits/sec	106 kbits/sec	212 kbits/sec	106-424 kbits/sec
Protocol	Specific Command set	Specific Command Set	FeliCa protocol	ISO 14443-4 ISO 7816-4 commands
Memory Size	Up to 1 KB	Up to 2 KB	Up to 1 MB	Up to 64KB
Cost (memory dependent)	Low	Low	Moderate	Moderate
Use cases	Tags with small memory for single application		Flexible tags with larger memory offering multi-application capabilities	

Figura 2.7: Comparativa de los distintos tipos de Tags.

- **Etiquetas semi-pasivas:** Las etiquetas semi-pasivas son un híbrido de las activas y las pasivas. Tienen una pequeña batería que se carga parcialmente cada vez que entra dentro del campo magnético de un lector.

2.2.8 Importancia de NFC

- **Alcance y disponibilidad:** NFC tiene el potencial para ser integrado en cada uno de los teléfonos del mundo. Mediante la integración de la tecnología NFC en móviles, los usuarios pueden obtener acceso a nuevos servicios a través de su teléfono.

- **Variedad de uso:** NFC puede ser usado para varias tareas, desde pago de bienes, hasta emparejamiento de los dispositivos para el intercambio de información o el descubrimiento de nuevos servicios.

- **Fácil de usar**: Porque NFC sólo requiere que dos dispositivos se toquen con el fin de comunicarse.

- **Seguridad**: NFC requiere de un usuario para activarlo manualmente o mantener su dispositivo móvil junto a otro móvil o junto a la estación NFC para activar un servicio o para compartir información. Al hacerlo la tecnología requiere del usuario para hacer una acción positiva que confirme la transacción o el intercambio. Además es posible construir múltiples niveles de seguridad en un dispositivo NFC.

- **Servicios de valor añadido**: NFC permite a los usuarios acceder a los servicios de valor añadido que de otro modo no están disponibles en una obtención de billetes o en un pago con tarjeta.

- **Infraestructura:** NFC es compatible con la estructura sin contactos. El despliegue de NFC es una extensión de los servicios que ya existen, pero es mayor con el elemento adicional de un interfaz de usuario del teléfono móvil y una conexión a Internet.

2.2.9 NFC Forum

El Near Field Communication (NFC) Forum es una asociación industrial sin animo de lucro fundada por NXP Semiconductors, Sony Corporation y Nokia para regular el uso de la interacción inalámbrica de corto alcance en la electrónica de consumo, dispositivos móviles y los PCs. [6]

Figura 2.8: Logo NFC Forum

Actualmente cuenta con más de 140 miembros entre los que podemos destacar:

Figura 2.9: Empresas que contribuyen al NFC Forum

El NFC Forum promueve la implantación y la estandarización de la Tecnología NFC como mecanismo para la interoperabilidad entre dispositivos y servicios.
Para conseguir esto, se encarga de:

- Desarrollar especificaciones basadas en estándares.
- Asegurarse del uso de las especificaciones del NFC Forum.
- Trabajar para que los productos con tecnología NFC cumplan con las especificaciones del NFC Forum.
- Educar a los consumidores y las empresas respecto de la Tecnología NFC.

El NFC Forum ha establecido un estándar en la que se registra un formato común para poder compartir datos entre los dispositivos NFC entre sí y/o entre los dispositivos y las etiquetas NFC.

❖ **NFC Data Exchange Format (NDEF):** Especifica un formato común y compacto para el intercambio de datos. [7]
❖ **NFC Record Type Definition (RTD):** Especifica tipos de registros estándar que pueden ser enviados en los mensajes intercambiados entre los dispositivos NFC. [8]
❖ **Smart Poster RTD:** Para posters que incorporen etiquetas con datos (URLs, SMSs o números de teléfono). [9]
❖ **Text RTD:** Para registros que solo contienen texto. [10]
❖ **Uniform Resource Identifier (URI) RTD:** Para registros que se refieren a un recurso de Internet. [11]

2.2.10 NFC API

De acuerdo con las normas ISO/IEC, los dispositivos NFC se pueden clasificar en tres modos de operación diferentes donde uno de los dispositivos funcionará en modo activo o pasivo: [12]

- **Card Emulation Mode**
  El dispositivo NFC actúa como una etiqueta pasiva (ver sección 2.5.3) al acercarse a un lector NFC externo. Este modo está estandarizado en la norma ISO/IEC 14443 y se puede utilizar, por ejemplo, en aplicaciones de pago.

- **Reader Mode**
  El dispositivo NFC actúa como un lector NFC activo en busca de etiquetas NFC pasivas. Este modo está estandarizado en la norma ISO/IEC 14443.

- **Peer to peer Mode**
  En este modo, un dispositivo NFC activo puede comunicarse con otro dispositivo NFC activo. Está estandarizado en la norma ISO/IEC 18092 y se puede utilizar, por ejemplo, para compartir parámetros de conguración de Bluetooth. Hasta la fecha, no existe API disponible para este modo.

Entre las distintas librerías que nos podemos encontrar tenemos:

- ***javax.microedition.contactless***
  - Proporciona funciones comunes a todos los dispositivos NFC.

- ***javax.microedition.contactless.ndef***
  - Proporciona funcionalidad para el intercambio de datos en formato NDEF (NFC Data Exchange Format) con otros dispositivos NFC.

- ***javax.microedition.contactless.rf***
  - Nos permite interactuar con dispositivos RF físicos.

- ***javax.microedition.contactless.sc***
  - Nos facilita la comunicación con tarjetas inteligentes externas (Smart Cards).

- ***javax.microedition.contactless.visual***
  - Proporciona formas de leer la información almacenada en códigos de barras (etiquetas visuales) y de generar dichas etiquetas

2.2.11 NFC en la vida real

Los usos más comunes de esta tecnología son pequeñas aplicaciones de pago como el transporte urbano, el aparcamiento público o para acceder a información.

En Japón, la operadora de telefonía móvilNTT DoCoMo ya probó el año pasado esta tecnología en el pago a través del móvil. Recientemente NTT DoCCoMo ha lanzado el nuevo servicio "DCMX mini" que permitirá a los usuarios comprar pasando su teléfono móvil NFC por lectores. La transacción se añadirá a la factura mensual del usuario. En Estados Unidos las estaciones de servicio de Exxon Mobile ofrecen ya este tipo de pagos.

Motorola ya ha anunciado que sus terminales incorporarán un chip NFC con funcionalidad de pago. Además, los teléfonos incorporarán una serie de características de seguridad para proteger los datos financieros y garantizar la seguridad de las transacciones financieras.

La industria de la música tampoco es ajena a esta tecnología. Philips,Visa y Universal Music Francia están trabajando en desarrollar un producto denominado “Smart Poster” que permitirá el pago de canciones desde cualquier lugar y dispositivo: desde un anuncio en una marquesina a una tienda de música. Posteriormente los usuarios podrán descargarse a través de Internet la canción comprada mediante este sistema.

En España, existen diferentes iniciativas piloto en la utilización de esta tecnología. La empresa Mobilpay, en colaboración con Indra, la Empresa Malagueña de Transportes, Oberthur y Orange, inauguraba en 2008 un sistema de pago mediante el móvil. Para ello, los dispositivos deben tener el chip NFC integrado en la SIM.

En concreto la tecnología NFC continúa creciendo con la aparición de distintos servicios en tarjetas de crédito y en pagos electrónicos. Según un reciente estudio de Abi Research, Japón y Corea del Sur lideran el mercado del pago a través de NFC.

2.2.12 NFC Aplicaciones

- Acceso al sistema público de transporte
- Pagos a través del móvil
- Control de acceso físico a lugares o recintos (como es el caso de este proyecto)
- Control de acceso a una red informática
- Tarjetas de fidelización o descuento
- Inclusión de información médica para uso en emergencias
- Apertura de vehículos
- Acceso a información desde cualquier soporte en la vía pública, museos,etc.

Figura 2.10 Aplicaciones NFC con el móvil

2.3 Bluetooth

2.3.1 Descripción

Con un gran número de dispositivos habilitados en la actualidad, incluyendo un alto porcentaje de los teléfonos móviles vendidos en el mundo, Bluetooth se presenta como la tecnología inalámbrica ideal para la conexión de dispositivos electrónicos.

Formalmente, Bluetooth es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia segura y globalmente libre.
Originalmente fue concebido como una alternativa inalámbrica a la transmisión de datos por cable mediante puerto serie. Posteriormente, se establecieron los siguientes objetivos con los que esta tecnología fue diseñada:

- Facilitar las comunicaciones entre equipos móviles y fijos.
- Eliminar cables y conectores entre dispositivos.
- Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la
- sincronización de datos entre equipos.
- Sistema universal, podrá operar en todo el mundo

Por su naturaleza, Bluetooth es un módulo de radio de baja potencia que puede integrarse en terminales de distinta naturaleza. En la actualidad, en los sectores de las telecomunicaciones y la informática personal, se incluye en aparatos como PDA, teléfonos móviles, ordenadores personales y portátiles, impresoras o cámaras digitales.
Además, esta tecnología permite la creación de redes de área personal conectando varios dispositivos mediante un enlace tanto de voz como de datos.

2.3.2 Etimología

El nombre procede del rey danés y noruego Harald Blåtand, cuya traducción al inglés sería Harold Bluetooth, conocido por buen comunicador y por unificar las tribus noruegas, suecas y danesas. La traducción textual al idioma español es "diente azul", aunque el término en danés era utilizado para denotar que Blåtand era de "tez oscura" y no de "dientes azules". [13]

Figura 2.11: Rey Harald.

La elección Bluetooth para denominar a esta nueva tecnología se debe a que pretende unir diferentes dispositivos como ordenadores, teléfonos móviles…
El logo de Bluetooth combina la representación de la runas nordicas *Hagalaz* (transcrito por “H”) y Berkana (transcrito por “B”) en un mismo símbolo.

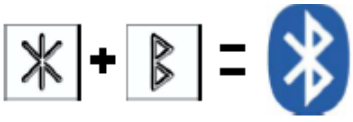


Figura 2.12: Logo Bluetooth.

2.3.3 Evolución

- **Bluetooth v.1.1:** en 1994, Ericsson inició un estudio para investigar la viabilidad de una nueva interfaz de bajo costo y consumo para la interconexión vía radio (eliminando así cables) entre dispositivos como teléfonos móviles y otros accesorios. El estudio partía de un largo proyecto que investigaba unos multicomunicadores conectados a una red celular, hasta que se llegó a un enlace de radio de corto alcance, llamado MC link. Conforme este proyecto avanzaba se fue haciendo claro que este tipo de enlace podía ser utilizado en un gran número de aplicaciones, ya que se basaba en un chip de radio.

- **Bluetooth v.1.2:** a diferencia de la 1.1, provee una solución inalámbrica complementaria para co-existir Bluetooth y Wi-Fi en el espectro de los 2.4 GHz, sin interferencia entre ellos. La versión 1.2 usa la técnica "Adaptive Frequency Hopping (AFH)", que ejecuta una transmisión más eficiente y un cifrado más seguro. Para mejorar las experiencias de los usuarios, la V1.2 ofrece una calidad de voz (Voice Quality - Enhanced Voice Processing) con menor ruido ambiental, y provee una más rápida configuración de la comunicación con otros dispositivos bluetooth dentro del rango del alcance, como pueden ser PDAs, HIDs (Human Interface Devices), ordenadores, Headsets, impresoras y teléfonos móviles.

- **Bluetooth v.2.0:** creada para ser una especificación separada, principalmente incorpora la técnica "Enhanced Data Rate" (EDR) que le permite mejorar las velocidades de transmisión en hasta 3Mbps a la vez que intenta solucionar algunos errores de la especificación 1.2.

- **Bluetooth v.2.1:** simplifica los pasos para crear la conexión entre dispositivos, además el consumo de potencia es 5 veces menor.

- **Bluetooth v3.0 (mediados 2009):** aumenta considerablemente la velocidad de transferencia. La idea es que el nuevo Bluetooth trabaje con WiFi, de tal manera que sea posible lograr mayor velocidad en los smartphones.

- **Bluetooth v4.0:** El SIG ha anunciado el lanzamiento de la especificación Bluetooth 4.0 para finales de 2010. Una de las principales mejoras es la llamada “energía baja” que permitirá que una gran variedad de dispositivos que antes eran incapaces de utilizar Bluetooth puedan ahora tomar ventaja de la conectividad que ofrece.

2.3.4 Características

Una frase que resume sus cualidades es: especificación abierta que utiliza en lugar de cables un enlace radio de corto alcance para comunicar dispositivos de voz y datos de forma simple y barata, pudiendo ser utilizada en cualquier parte del mundo.
Sin embargo, de la página oficial del SIG [sección 2.3.6], se obtiene una descripción más técnica de sus características mostradas a continuación:

- La tecnología inalámbrica Bluetooth está orientada a aplicaciones de voz y datos.
- Bluetooth funciona en la banda de frecuencia de 2.4 GHz, que no precisa de ninguna licencia.
- Bluetooth tiene un radio de acción de 10 o 100 metros dependiendo de la clase del dispositivo Bluetooth. La máxima velocidad de transmisión es de 3 Mbps.
- Los objetos sólidos no suponen ningún obstáculo para la tecnología inalámbrica Bluetooth.
- Con Bluetooth tampoco es necesario que los dispositivos estén situados en la misma línea de visión, es decir, orientados uno frente a otro, ya que se transmite en todas direcciones.
- La seguridad siempre ha sido una de las prioridades en el desarrollo de la tecnología Bluetooth y continúa siéndolo. La especificación Bluetooth ofrece tres modos de seguridad.
- El coste de los chips Bluetooth es inferior a tres dólares estadounidenses.

2.3.5 Comparación con otras tecnologías

La siguiente tabla muestra una comparación de las características técnicas de las diferentes tecnologías inalámbricas del mercado. En ella se aprecian los bajos valores de la tecnología Bluetooth en los campos de precio, consumo de energía y alcance. [14]

Característica	ZigBee	Bluetooth	802.11b	802.11g	802.11a	802.11n	UWB
Rendimiento (Mbps)	0.03	1-3	11	54	54	200	200
Alcance máximo (m)	23	10	60	60	45	45	10
Energía (mW)	30	100	750	1000	1500	2000	400
Ancho de banda (MHz)	0.6	1	22	20	20	40	500
Precio ($)	2	3	5	9	12	20	7

Tabla 2.2: Comparativa de Bluetooth con otras tecnologías.

2.3.6 Bluetooth SIG

El grupo de interés especial (SIG) de Bluetooth Es una asociación privada sin ánimo de lucro con sede en Bellevue, Washington. A fecha de septiembre de 2007, el SIG estaba formado por más de 9000 compañías de telecomunicaciones, informática, automovilismo, música, textil, automatización industrial y tecnologías de red. Tiene pequeños grupos de personal dedicado al grupo en Hong Kong, Suecia y Estados Unidos. Los miembros del SIG dirigen el desarrollo de la tecnología inalámbrica Bluetooth, además de implementar y comercializar la tecnología en sus productos. El Bluetooth SIG por sí mismo no fabrica ni vende dispositivos Bluetooth. [15]

Cualquier compañía que incorpora la tecnología inalámbrica Bluetooth en productos, utilizando la tecnología para ofrecer bienes y servicios debe convertirse en un miembro de Bluetooth SIG. Hay tres niveles de socios corporativos por un total de más de 12.000 miembros. Estos miembros son los más activos en el **SIG** y tienen influencia tanto sobre las orientaciones estratégicas y tecnológicas de Bluetooth en su conjunto. Los miembros actuales son el promotor:

- Ericsson (miembro fundador)
- Intel Corporation (miembro fundador)
- Lenovo
- Microsoft (desde 2008)
- Motorola(desde 1999)
- Nokia
- Toshiba (miembro fundador)

Figura 2.13: Logo de SIG.

2.3.7 Conexiones Bluetooth

Bluetooth proporciona una conexión punto a punto o punto a multipunto. En este último caso, el canal es compartido por varias unidades Bluetooth. Dos o más unidades que comparten el mismo canal forman una *piconet*. [16]
Una unidad actúa como maestro de la *piconet*, mientras que el resto de unidades actúan como esclavos. La norma permite a cualquier equipo asumir cualquiera de los dos papeles, incluso un equipo puede actuar como maestro en un enlace y como esclavo en otro. El maestro es el que se encarga de la sincronización en la *piconet*.

El número máximo de elementos activos en una *piconet* es de ocho. Como uno asume el rol de maestro, como máximo pueden existir activos siete esclavos. Esto influye directamente en el sistema de la aplicación del proyecto estableciendo este número como el máximo de sensores que se pueden conectar. Además, la *piconet* puede contar con muchos más esclavos en estado de espera, que se mantienen sincronizados con el maestro. El acceso al canal es controlado por el maestro.

Figura 2.14: Modo de funcionamiento maestro-esclavos.

Múltiples *piconets* con áreas de cobertura solapadas forman una *scatternet*. Cada *piconet* sólo puede tener un maestro. Sin embargo, los esclavos pueden participar en *piconets* diferentes utilizando multiplexación por división en el tiempo. Las *piconets* que forman una *scatternet* [17] no estarán sincronizadas en el tiempo o en frecuencia. Cada una tiene su propio canal con sus correspondientes saltos en frecuencia. En la siguiente figura se muestran ejemplos de estas estructuras:

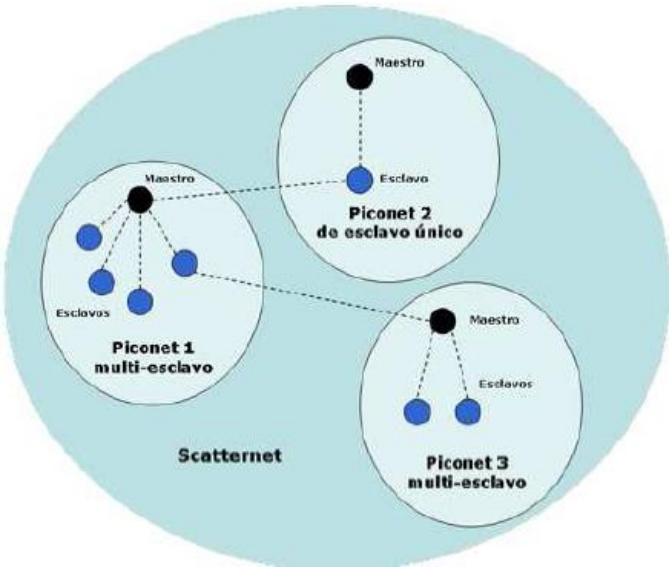


Figura 2.15: Piconets unidas formando un Scatteret.

Al establecer la conexión existen dos procedimientos, según la información que se tenga:

- Si no se conoce nada del aparato remoto debe seguirse primero un procedimiento de búsqueda, denominado inquiry, y después otro de asociación, llamado page.
- Si se conocen detalles del aparato remoto (dirección física y fase de reloj) tan sólo es necesario llevar a cabo el procedimiento de asociación page.

El procedimiento de búsqueda permite a un dispositivo descubrir qué otros equipos se encuentran en su área de cobertura y conseguir la información necesaria para conectarse con ellos. El maestro pregunta si hay dispositivos presentes enviando mensajes IAC si quiere recibir respuesta de cualquier tipo de dispositivo o con un mensaje GIAC si pregunta por algún dispositivo en particular. Un terminal que quiera ser descubierto escuchará el canal y responderá con el valor de su dirección y reloj. Posteriormente, se establece la conexión al conocerse la dirección del dispositivo remoto.

Otro aspecto importante en las comunicaciones inalámbricas es la seguridad. Por ello el estándar de Bluetooth define algoritmos de autenticación y cifrado en el establecimiento de los enlaces.

2.3.8 Arquitectura

Uno de los principales objetivos la tecnología Bluetooth es conseguir que aplicaciones de diferentes fabricantes mantengan una comunicación fluida. Para ello, los distintos sistemas deberán estar implementados sobre una misma estructura de protocolos.

• **Pila de Protocolos**

En la Figura 2.16 se muestra la pila de protocolos completa definida en la especificación Bluetooth.

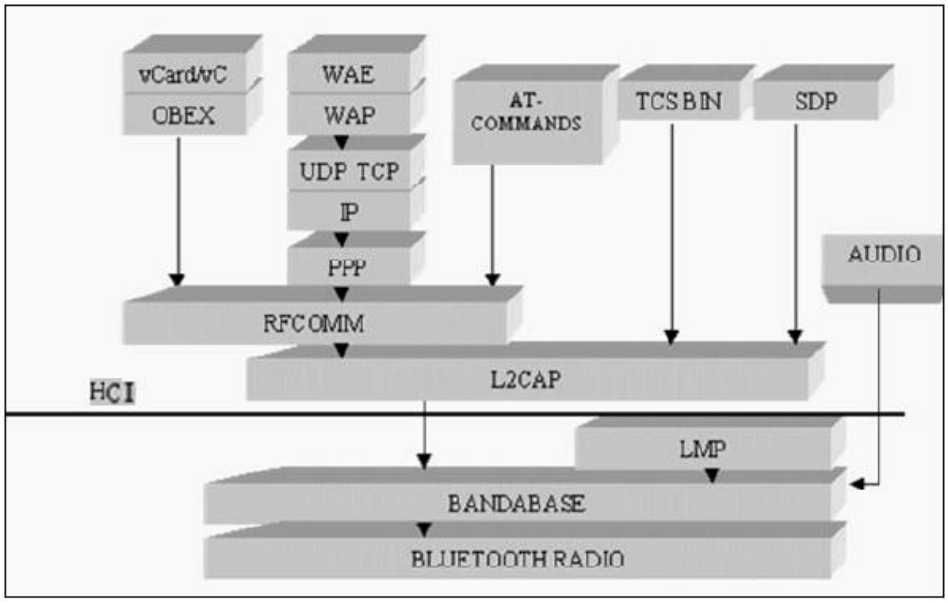


Figura 2.16: Pila de Protocolos Bluetooth.

En el diseño la arquitectura se ha intentado maximizar el número de aplicaciones que puedan implementarse con Bluetooth. Para ello, se han reutilizado protocolos existentes en capas superiores, siendo específicos de Bluetooth los de nivel más bajo. Normalmente, las aplicaciones no hacen uso de todos los protocolos. Sin embargo, la especificación es abierta y permite a los fabricantes desarrollar sus propios protocolos de aplicación para crear más aplicaciones que usen Bluetooth.

La pila de la figura está compuesta por numerosos protocolos que pueden ser clasificados según su propósito en los siguientes cuatro grupos:

- Protocolos centrales de Bluetooth: Banda Base, LMP, L2CAP, SDP.
- Protocolos de reemplazo de cable: RFCOMM.
- Protocolos de control de telefonía: TCS BIN, AT-Comands.
- Protocolos adaptados: PPP, UDP/TCP/IP, OBEX, WAP, vCard, vCal, WAE.

Los protocolos centrales de Bluetooth son aquellos que han sido desarrollados por el SIG en la mayoría de los dispositivos y realizan las funciones de los niveles físico, enlace y red del modelo de referencia OSI. En cambio, los protocolos de reemplazo de cable, control de telefonía y protocolos adoptados constituyen los orientados a aplicación y han de permitir que las aplicaciones funcionen sobre los protocolos centrales de Bluetooth.

La capa de radio es la inferior del estándar y puede ser identificada con el nivel físico. Es la responsable de la transmisión de los datos. Los dispositivos Bluetooth funcionan en la banda de 2,4 GHz que no requieren licencia. Los dispositivos usan un transmisor de salto de frecuencia para contrarrestar interferencias y pérdida de intensidad.

La banda base podría englobarse también dentro de la capa física de la torre OSI. Su función principal es controlar los canales y enlaces físicos. Además es la encargada de realizar funciones de sincronización, codificación, decodificación, control de errores y seguridad de bajo nivel. Soporta enlaces de dos tipos SCO y ACL. El primero es un enlace simétrico punto a punto mientras que el segundo es un enlace punto-multipunto.

A continuación se hace una descripción de los protocolos más importantes:

- **L2CAP o Protocolo de Adaptación y Control de Enlace Lógico:** está implementado sobre el protocolo de Banda Base. Proporciona servicios de datos orientados o no a conexión para los protocolos de capas superiores. Se ocultan a las aplicaciones de alto nivel las particularidades del nivel de enlace y físico. [18]
- **SDP o Protocolo de Descubrimiento de Servicios**: proporciona a las aplicaciones una forma de descubrir los servicios que están disponibles y determinar sus características. Está montado sobre enlaces L2CAP. Una vez que se ha establecido un enlace L2CAP entre dos dispositivos, puede usarse este protocolo para encontrar servicios y conectarse a ellos. Los servicios SDP están compuestos por una serie de atributos. Cada clase de servicio tiene asignado un identificador único, el UUID. [19]
- **RFCOMM o Protocolo de Emulación de Puerto Serie**: se encarga de la emulación de los puertos serie (RS-232) sobre el protocolo L2CAP. Del mismo modo que los puertos serie tienen nueve circuitos, que se usan para transferir datos y señalización, RFCOMM puede emular sus configuraciones y estados. [20]
- **OBEX o Protocolo de Intercambio de Objetos**: está desarrollado para el intercambio simple de objetos. Usa una estructura cliente/servidor. En su definición incluye un modelo de representación de objetos, un formato estándar para transmitirlos y un protocolo de sesión para el intercambio de peticiones y respuestas entre dispositivos. [21]

2.3.9 Perfiles Bluetooth

El estándar Bluetooth se diseñó con el objetivo de ser usado por un amplio conjunto de fabricantes y ser implementado en distintos campos. Para evitar distintas interpretaciones del estándar, el SIG ha definido perfiles Bluetooth.

Un **perfil Bluetooth** es la especificación de una interfaz de alto nivel para su uso entre dispositivos Bluetooth. Para utilizar una cierta tecnología Bluetooth un dispositivo deberá soportar ciertos perfiles.Los perfiles son descripciones de comportamientos generales que los dispositivos pueden utilizar para comunicarse, formalizados para favorecer un uso unificado. La forma de utilizar las capacidades de Bluetooth se basa, por tanto, en los perfiles que soporta cada dispositivo. Un perfil define una selección de mensajes y protocolos necesarios para la implementación de una determinada aplicación.

La definición de cada perfil Bluetooth contiene, como mínimo, información acerca de:

- Dependencias con otros perfiles.
- Sugerencias acerca del formato de las interfaces de usuario.
- Partes específicas del protocolo de pila usado por el perfil.

Para llevar a cabo esta tarea cada perfil utiliza opciones particulares y parámetros en cada capa de la pila.

La tecnología Bluetooth describe un largo conjunto de perfiles que ilustran diferentes tipos de escenarios donde pueden ser utilizados.

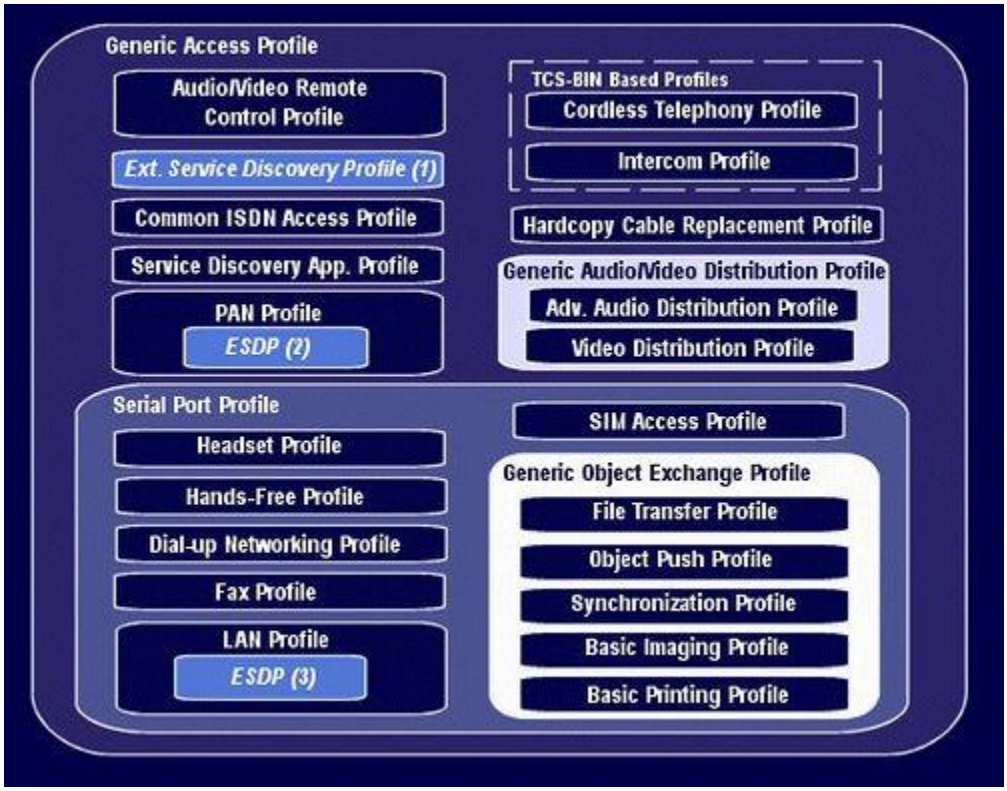


Figura 2.17: Perfiles Bluetooth

2.3.10 Bluetooth API

Entre las distintas librerías que nos podemos encontrar tenemos:

- ***javax.bluetooth.DeviceClass***
  - Define valores para el tipo de dispositivo y los tipos de servicio del dispositivo.

- ***javax.bluetooth.DataElement***
  - Contirne varios tipos de datos que un atributo de un servicio puede tomar. Pueden ser: enteros sin sigo, String, boolean o UUID.
  - Permite crear o recuperar el valor de un atributo de servicio.

- ***javax.bluetooth.ServiceRecord***
  - Define un ID, que es un elemento sin signo de 16bits y un *valor* que es un DataElement. Además, esta interfaz contiene el método *populateRecord()* que sirve pare recuperar atributos de los servicios deseados.

- ***javax.bluetooth.DiscoveryAgent***
  - Proporciona métodos para el descubrimiento de servicios y dispositivos, como *startInquiry()* o *retrieveDevices().*
  - También proporciona método para detener el descubrimiento, como *cancelInquiry()*

- ***javax.bluetooth.DiscoveryListener***
  - Esta interfaz permite a una aplicación especificar un "event listener" que responda a eventos de descubrimiento de servicios y dispositivos. El método *servicesDiscovered()* se llama cuando los servicios se descubren. El método *serviceSearchCompleted* se llama cuando la búsqueda de dispositivos se completa o se cancela.

- ***javax.bluetooth.LocalDevice***
  - Proporciona acceso y control sobre el dispositivo Bluettoth local. Esta diseñada para cumplir completamente los requisitos definidos en la especificación Bluetooth.

- ***javax. bluetooth. RemoteDevice***
  - Proporciona información básica sobre un dispositivo remoto, incluyendo la dirección bluetooth del dipositivo y su nombre amigable (el nombre bluetooth del dispositivo).

- ***javax. bluetooth. UUID***
  - Encapsula enteros sin signo de 16,32 o 128bits. Se utiliza para representar un identificador único y universal utlizado como valor para un atributo de un servicio.

2.3.11 Bluetooth en la vida real

Una de las primeras compañías en lanzar un producto Bluetooth ha sido Ericsson. Alianzas como las de Nokia y Fuji permitirán a los propietarios de cámaras digitales hacer fotos para luego transmitirlas a través del móvil a la impresora situada en casa o al disco duro del ordenador. Mientras, compañías como Motorola y JVC desarrollan continuamente tecnologías aún más avanzadas que harán estos avances extensibles al vídeo o al DVD. La compañía Sony ha hecho posible la implantación de microchips Bluetooth en toda su gama de productos. En sólo un par de años caminaremos escuchando música en un reproductor MP3 mientras descargamos nuevas canciones y actualizamos el repertorio musical a través del móvil. La utilidad de Bluetooth sólo está delimitada por la imaginación de los ingenieros y los usuarios.

Hoy en día, la aparición del Bluetooth permite cambiar radicalmente la forma en la que los usuarios interactúan con los teléfonos móviles y otros dispositivos, dando lugar así a un gran numero de aplicaciones y usos de esta tecnología.

Desde el punto de vista de los usuarios, Bluetooth supone una tecnología que permite una comunicación fácil, instantánea, rápida, en cualquier lugar y que además su coste es bajo; sin olvidar su impacto en la forma de realizar los procesos, al sustituir los medios convencionales y posibilitar nuevos negocios y aplicaciones.

La aplicación de esta tecnología se puede percibir desde la implementación de una red inalámbrica hasta la posibilidad de transferir una fotografía de una cámara a un móvil para enviarla por correo electrónico o transferirla a la impresora para imprimirla en ausencia de cables.

Desde el punto de vista profesional, la aplicación más práctica, es la posibilidad de montar una red inalámbrica en salas o entornos que ofrezcan dificultades para montar una LAN convencional. Para ello se utiliza un punto de acceso y cada puesto lleva instalado una PC Card con esta tecnología.

2.3.12 Bluetooth Aplicaciones

- Transferencia de archivos
- Transferencia de archivos
- Escritorio Inalámbrico
- Conexión a Internet
- Acesso Inalámbrico a LAN
- Sincronización Automática
- Dispositivo Manos Libres Inalámbrico.

Figura 2.18: Transferencia de datos mediante Bluetooth a distintos dipositivos.

Como se puede apreciar, las aplicaciones de Bluetooth son casi infinitas y permiten cambiar radicalmente la forma en la que los usuarios interactúancon los teléfonos móviles y otros dispositivos. Entre otras aplicaciones, Bluetooth permite conectar cámaras de vigilancia, servir con mandos a distancia, permite utilizar un teléfono celular como inalámbrico, para abrir puertas, conectar electrodomésticos, pasar ficheros MP3 del móvil al PC, y por supuesto, para conectar todo tipo de dispositivos a Internet, formando puntos de acceso. En definitiva, Bluetooth se esta convirtiendo en una tecnología de uso cotidiano.

2.4 Java

Java es un lenguaje de programación muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Fue desarrollado por Sun Microsystems y está enfocado a cubrir las necesidades tecnológicas.

Una de las principales características es que es un lenguaje independiente de la plataforma, es decir, un programa en Java podrá funcionar en cualquier ordenador indistintamente del sistema operativo. Es una ventaja significativa para los desarrolladores de software, pues antes tenían que hacer un programa para cada sistema operativo, por ejemplo Windows, Linux, Apple, etc. Esto lo consigue porque se ha creado una Máquina de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java y posibilita que este último se entienda perfectamente. Otra de las ventajas es que Java está desarrollándose para distintos tipos de dispositivos además del ordenador como móviles, agendas y en general para cualquier cosa que se le ocurra a la industria.

La interoperatividad que proporciona la plataforma Java es también muy útil para los usuarios de dispositivos móviles. Las aplicaciones creadas con los APIs (Application Programming Interface) estándar de Java deben ejecutarse en todos los dispositivos compatibles, no importa quien los fabricó.

Existen básicamente tres plataformas:

- Java Platform, Standard Edition (Java SE). [22]
- Java Platform, Enterprise Edition (Java EE). [23]
- Java Platform, Micro Edition (Java ME). [24]

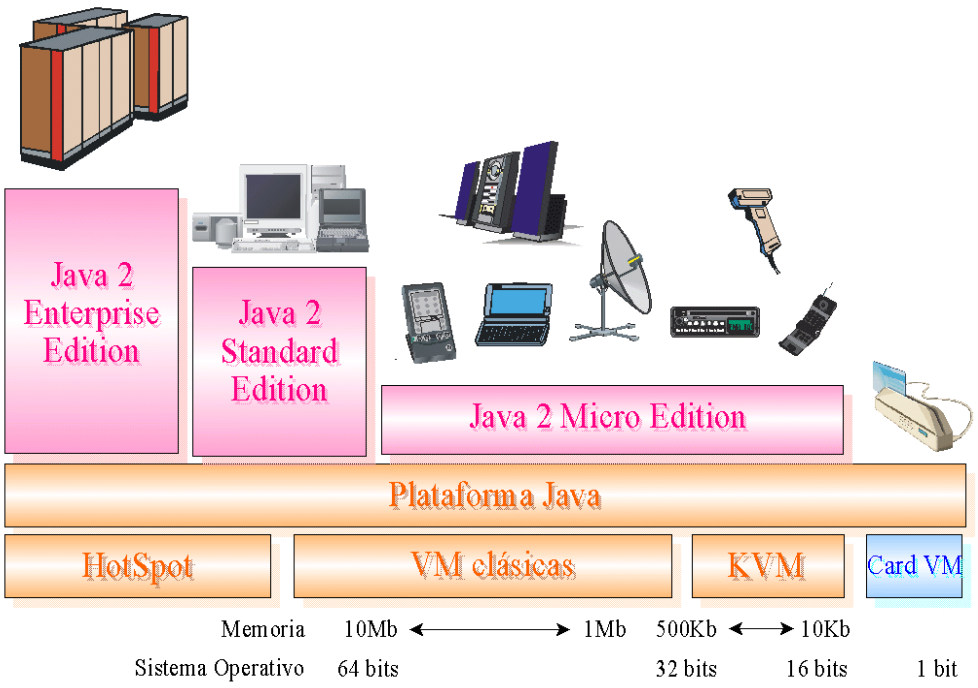


Figura 2.19: Arquitectura de la plataforma Java 2 de Sun

2.4.1 J2ME

Esta versión está enfocada a la aplicación de la tecnología Java en dispositivos electrónicos con capacidades muy reducidas, como teléfonos móviles, PDAs o electrodomésticos.

2.4.1.1 Diferencias con J2EE y J2SE

Posee componentes que la diferencian de las otras versiones, como una máquina virtual denominada KVM (Kilo Virtual Machine, debido a que requiere sólo unos pocos Kilobytes de memoria para funcionar) en vez de la JVM clásica, inclusión de un pequeño y rápido recolector de basura y otras diferencias.

J2ME contiene una mínima parte de las APIs de Java. Esto es debido a que la edición estándar de APIs de Java ocupa 20 Mb, y los dispositivos pequeños disponen de una cantidad de memoria mucho más reducida. La parte del API que se mantiene fija forma parte de lo que se denomina "configuración".

J2ME representa una versión simplificada de J2SE. Sun separó estas dos versiones ya que J2ME estaba pensada para dispositivos con limitaciones de proceso y capacidad gráfica. También separó J2SE de J2EE porque este último exigía unas características muy pesadas o especializadas de E/S, trabajo en red, etc. Por tanto, separó ambos productos por razones de eficiencia. Hoy, J2EE es un superconjunto de J2SE pues contiene toda la funcionalidad de éste y más características, así como J2ME es un subconjunto de J2SE (excepto por el paquete javax.microedition) ya que contiene limitaciones con respecto a J2SE.

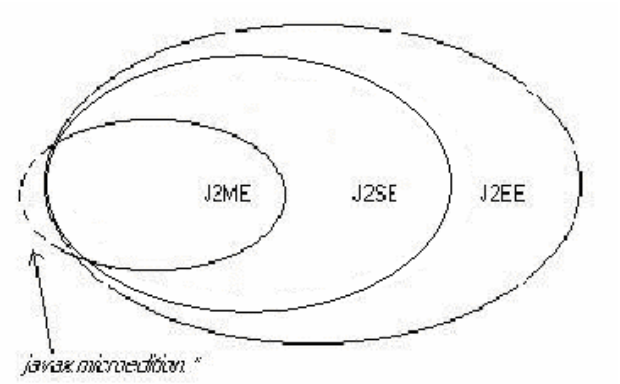


Figura 2.20: Relación entre las APIs de la plataforma Java.

2.4.1.2 Componenetes

- **Configuraciones:** conjunto de clases básicas orientadas a conformar las implementaciones para dispositivos de características específicas. Son:
  - **Connected Limited Device Configuration (CLDC)** enfocada a dispositivos con restricciones de procesamiento y memoria. [25]
  - **Connected Device Configuration (CDC)** enfocada a dispositivos con más recursos. [26]

- **Perfiles**: bibliotecas Java de clases específicas orientadas a implementar funcionalidades de más alto nivel para familias específicas de dispositivos.

Un entorno de ejecución determinado de J2ME se compone de una selección de:

a) Máquina virtual.

b) Configuración.

c) Perfil.

d) Paquetes Opcionales.

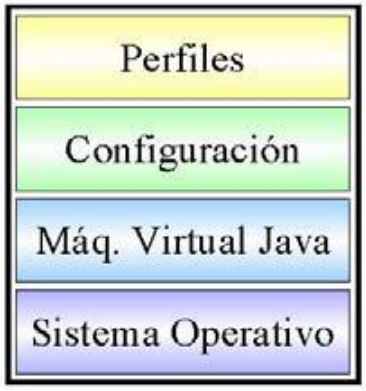


Figura 2.21: Entorno de ejecución

2.4.1.3 Máquinas Virtuales J2ME

Una máquina virtual de Java (JVM) es un programa encargado de interpretar código intermedio (bytecode) de los programas Java precompilados a código máquina ejecutable por la plataforma, efectuar las llamadas pertinentes al sistema operativo y observar las reglas de seguridad y corrección de código definidas para el lenguaje Java. De esta forma, la JVM proporciona al programa Java independencia de la plataforma con respecto al hardware y al sistema operativo subyacente.

La VM (Virtual Machine) de la configuración CLDC se denomina KVM [27] y la de la configuración CDC se denomina CVM [28].

- **KVM**

Se corresponde con la Máquina Virtual más pequeña desarrollada por Sun. Su nombre KVM proviene de Kilobyte (haciendo referencia a la baja ocupación de memoria, entre 40Kb y 80Kb). Se trata de una implementación de Máquina Virtual reducida y orientada a dispositivos con bajas capacidades computacionales y de memoria.

La KVM está escrita en lenguaje C, aproximadamente unas 24000 líneas de código, y fue diseñada para ser:

- Pequeña, con una carga de memoria entre los 40Kb y los 80 Kb, dependiendo de la plataforma y las opciones de compilación.
- Alta portabilidad.
- Modulable.
- Lo más completa y rápida posible.

Sin embargo, la baja ocupación de memoria hace que posea algunas limitaciones con respecto a la clásica *Java Virtual Machine* (JVM).

❖ **CVM**

La CVM (Compact Virtual Machine) ha sido tomada como Máquina Virtual Java de referencia para la configuración CDC y soporta las mismas características que la Máquina Virtual de J2SE. Está orientada a dispositivos electrónicos con procesadores de 32 bits de gama alta y en torno a 2Mb o más de memoria RAM. Las características que presenta esta Máquina Virtual son:

- Sistema de memoria avanzado.
- Tiempo de espera bajo para el recolector de basura.
- Separación completa de la VM del sistema de memoria.
- Recolector de basura modularizado.
- Portabilidad.
- Rápida sincronización.
- Ejecución de las clases Java fuera de la memoria de sólo lectura (ROM).
- Soporte nativo de hilos.
- Baja ocupación en memoria de las clases.
- Proporciona soporte e interfaces para servicios en Sistemas Operativos de
- Tiempo Real.
- Conversión de hilos Java a hilos nativos.
- Soporte para todas las características de Java2 v1.3 y librerías de seguridad, referencias débiles, Interfaz Nativa de Java (JNI), invocación remota de métodos (RMI), Interfaz de depuración de la Máquina Virtual (JVMDI).

2.4.1.4 Configuraciones

Una configuración es el conjunto mínimo de APIs Java que permiten desarrollar aplicaciones para un grupo de dispositivos. Éstas APIs describen las características básicas, comunes a todos los dispositivos:

• Características soportadas del lenguaje de programación Java.
• Características soportadas por la Máquina Virtual Java.
• Bibliotecas básicas de Java y APIs soportadas.

Existen dos configuraciones en J2ME:

❖ **Connected Device Configuration, CDC.**

Orientado dispositivo con cierta capacidad computacional y de memoria. Usa la Máquina Virtual CVM.

La CDC está enfocada a dispositivos con las siguientes capacidades:

- Procesador de 32 bits.
- Disponer de 2 Mb o más de memoria total, incluyendo memoria RAM y
- ROM.
- Poseer la funcionalidad completa de la Máquina Virtual Java2.
- Conectividad a algún tipo de red.

Las peculiaridades de la CDC están en el paquete javax.microedition.io, que incluye soporte para comunicaciones http y basadas en datagramas.

Nombre de Paquete CDC	Descripción
java.io	Clases e interfaces estándar de E/S.
java.lang	Clases básicas del lenguaje.
java.lang.ref	Clases de referencia.
java.lang.reflect	Clases e interfaces de reflection.
java.math	Paquete de matemáticas.
java.net	Clases e interfaces de red.
java.security	Clases e interfaces de seguridad
java.security.cert	Clases de certificados de seguridad.
java.text	Paquete de texto.
java.util	Clases de utilidades estándar.
java.util.jar	Clases y utilidades para archivos JAR.
java.util.zip	Clases y utilidades para archivos ZIP y comprimidos.
javax.microedition.io	Clases e interfaces para conexión genérica CDC.

Tabla 2.3: Librerías de configuración CDC

- **Connected Limited Device Configuration, CLDC**.

La CLDC está orientada a dispositivos dotados de conexión y con limitaciones en cuanto a capacidad gráfica, cómputo y memoria. Un ejemplo de estos dispositivos son: teléfonos móviles, PDAs, etc. Algunas de estas limitaciones vienen dadas por el uso de la KVM, necesaria al trabajar con la CLDC debido a su pequeño tamaño. Los dispositivos que usan CLDC deben cumplir los siguientes requisitos:

- Disponer entre 160 Kb y 512 Kb de memoria total disponible. Como mínimo se debe disponer de 128 Kb de memoria no volátil para la Máquina Virtual Java y las bibliotecas CLDC, y 32 Kb de memoria volátil para la Máquina Virtual en tiempo de ejecución.
- Procesador de 16 o 32 bits con al menos 25 Mhz de velocidad.
- Ofrecer bajo consumo, debido a que estos dispositivos trabajan con suministro de energía limitado, normalmente baterías.
- Tener conexión a algún tipo de red, normalmente sin cable, con conexión intermitente y ancho de banda limitado (unos 9600 bps).

Nombre de paquete CLDC	Descripción
java.io	Clases y paquetes estándar de E/S. Subconjunto de J2SE.
java.lang	Clases e interfaces de la Máquina Virtual. Subconj. de J2SE.
java.util	Clases, interfaces y utilidades estándar. Subconj. de J2SE.
javax.microedition.io	Clases e interfaces de conexión genérica CLDC

Tabla 2.4: Librerías incluidas en la CLDC.

En cualquier caso, una determinada Configuración no se encarga del mantenimiento del ciclo de vida de la aplicación, interfaces de usuario o manejo de eventos, sino que estas responsabilidades caen en manos de los Perfiles.

2.4.1.5 Perfiles

Un perfil es un conjunto de APIs orientado a un ámbito de aplicación determinado. Los perfiles identifican un grupo de dispositivos por la funcionalidad que proporcionan (electrodomésticos, teléfonos móviles, etc.) y el tipo de aplicaciones que se ejecutarán en ellos. Las librerías de la interfaz gráfica son un componente muy importante en la definición de un perfil. Se pueden encontrar grandes diferencias entre interfaces, desde el menú textual de los teléfonos móviles hasta los táctiles de los PDAs.

El perfil establece unas APIs que definen las características de un dispositivo, mientras que la configuración hace lo propio con una familia de ellos. Esto hace que a la hora de construir una aplicación se cuente tanto con las APIs del perfil como de la configuración. Tenemos que tener en cuenta que un perfil siempre se construye sobre una configuración determinada. De este modo, podemos pensar en un perfil como un conjunto de APIs que dotan a una configuración de funcionalidad específica. [29]

Existen unos perfiles que construiremos sobre la configuración CDC y otros que construiremos sobre la CLDC.

Para la configuración CDC tenemos los siguientes perfiles:

- **Foundation Profile**
- **Personal Profile**
- **RMI Profile.**

Para la configuración CLDC tenemos los siguientes:

- **PDA Profile.**
- **Mobile Information Device Profile (MIDP).**

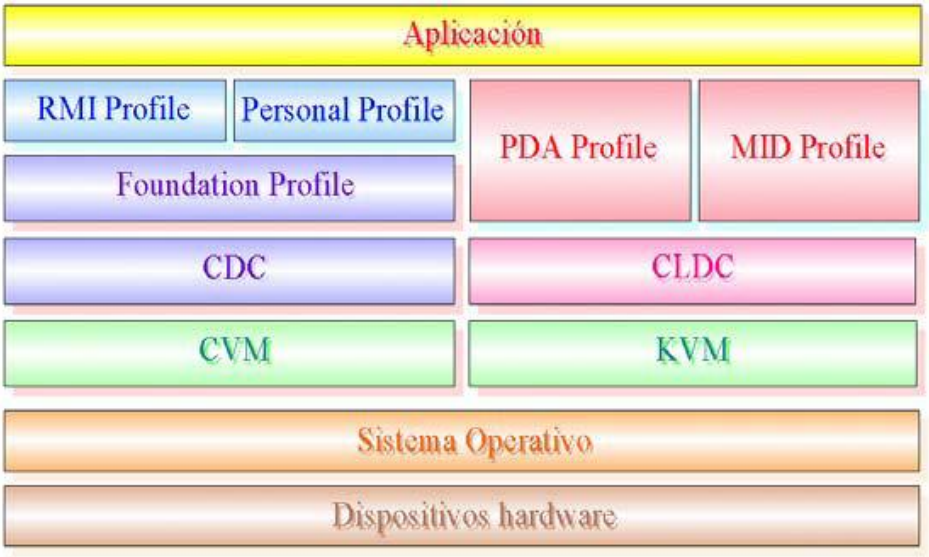


Figura 2.22: Arquitectura del entorno de ejecución de J2ME.

2.4.1.5.1 MIDP

MIDP es el acrónimo de "Perfil para dispositivos de información móvil" (Mobile Information Device Profile) y nos proporciona un perfil que se apoya en CLDC y que nos va a proporcionar los paquetes y clases necesarios para el desarrollo de nuestras aplicaciones. MIDP está orientado principalmente a teléfonos móviles, aunque existe una implementación para PalmOS (versión 3.5 y superiores) y PocketPC, por lo que es también utilizable en casi cualquier PDA. [30]

Este perfil está orientado para dispositivos con las siguientes características:

- Reducida capacidad computacional y de memoria.
- Conectividad limitada (en torno a 9600 bps).
- Capacidad gráfica muy reducida (mínimo un display de 96x54 pixels monocromo).
- Entrada de datos alfanumérica reducida.
- 128 Kb de memoria no volátil para componentes MIDP.
- 8 Kb de memoria no volátil para datos persistentes de aplicaciones.
- 32 Kb de memoria volátil en tiempo de ejecución para la pila Java.

Los tipos de dispositivos que se adaptan a estas características son: teléfonos móviles, buscapersonas (pagers) o PDAs de gama baja con conectividad.

El perfil MIDP establece las capacidades del dispositivo, por lo tanto, especifica las APIs relacionadas con:

- La aplicación (semántica y control de la aplicación MIDP).
- Interfaz de usuario.
- Almacenamiento persistente.
- Trabajo en red.
- Temporizadores.

2.4.1.5.2 Versiones

- MIDP 1.0 (JSR 37) – Publicada el 19Septiemnre de 2000
- MIDP 2.0 (JSR 118) - Publicada el Approved 20 de Noviembre de 2002
- MIDP 3.0 (JSR 271) - Publicada el 9 de Diciembre de 2009

2.5 JAVACARD

Java Card es una tecnología que permite ejecutar de forma segura pequeñas aplicaciones Java (applets) que emplean la tecnología Java, en tarjetas inteligentes y similares dispositivos empotrados. JavaCard es compatible con estándares de tarjetas inteligentes.

Proporciona al usuario la capacidad de programar aplicaciones que se ejecutan en la tarjeta de modo que ésta tenga una funcionalidad práctica en un dominio de aplicación específico (pe. identificación, pago, etc.). Esta tecnología se usa ampliamente en las tarjetas SIM (utilizadas en teléfonos móviles GSM) y en tarjetas monedero electrónico.

La primera tarjeta Java fue presentada en 1997 por varias empresas entre las que estaban Axalto (división de tarjetas de Schlumberger) y Gemplus. Los productos Java Card están basados en la Plataforma Java Card cuyas especificaciones han sido desarrolladas por Sun Microsystems.

Para el caso de este proyecto la versión de Java Card utilizada es la Java Card v2.2.1 [31] que consta de tres documentos:

1. Especificación de la maquina virtual (JCVM) proporciona el conjunto de instrucciones de la JCVM (Java Card Virtual Machine), soporte para el uso de un subconjunto del lenguaje Java, y los formatos de archivo utilizados para instalar librerías y applets Java Card en dispositivos habilitados.

2. Especificación del entorno en tiempo de ejecución (JCRE) define el comportamiento necesario del entorno en tiempo de ejecución (RE) en cualquier aplicación de la tecnología Java Card. El RE incluye la implementación de la JCVM, las clases del API Java Card y servicios de soporte en tiempo de ejecución, tales como, la selección y deselección de applets.

3. Interfaz de programación de aplicaciones (API) que describe los paquetes y clases Java Card para programar aplicaciones en tarjetas inteligentes.

2.5.1 Características

- **Portabilidad**

Java Card tiene por objeto la definición de un estándar de tarjeta inteligente que permite al applet funcionar en diferentes tarjetas inteligentes, de forma muy parecida a cómo un applet de Java se ejecuta en diferentes ordenadores. Al igual que en Java, esto se consigue utilizando la combinación de una máquina virtual (la Máquina virtual de Java Card), y unas librerías cuya API está especificada. La portabilidad, en todo caso, sigue siendo obviada en muchos casos por cuestiones de tamaño de la memoria, el rendimiento

y tiempo de ejecución (por ejemplo para los protocolos de comunicación o algoritmos criptográficos).

❖ **Seguridad**

La tecnología Java Card fue desarrollada originalmente con el propósito de asegurar la información sensible almacenada en las tarjetas inteligentes. La seguridad está determinada por diversos aspectos de esta tecnología:

- **Applet**. El applet es una máquina de estados que sólo procesa los comandos recibidos a través del dispositivo lector enviando y respondiendo con códigos de estado y datos.

- **Separación de applets**. Las distintas aplicaciones están, además, separadas unas de otras por unos cortafuegos que limita el acceso y control de los elementos de datos de un subprograma a otro.

- **Encapsulación de datos**. Los datos se almacenan en la aplicación y las aplicaciones Java Card se ejecutan en un entorno aislado (la tarjeta de Java VM), separada del sistema operativo y del equipo en que se lee la tarjeta.

- **Criptografía**. En esta plataforma están implementados los algoritmos criptográficos más comúnmente utilizados como DES, 3DES, AES, RSA (incluyendo el uso de criptografía de curva elíptica). Otros servicios como la firma electrónica o la generación de claves de intercambio también están soportados.

2.5.2 JavaCard frente a Java

Debido a las limitaciones de memoria y procesamiento de una tarjeta inteligente, la tecnología Java Card es más reducida que Java. En esta sección se comentarán las principales diferencias que existen entre ambas tecnologías.

❖ **Lenguaje**

A nivel de lenguaje, Java Card es un subconjunto de Java: todas las construcciones del lenguaje Java Card existen en Java y se comportan de la misma manera. Esto va hasta el punto de que, como parte de un ciclo estándar de desarrollo, un applet Java Card se compila en un archivo de clase Java (.class) por un compilador Java normal, sin ningún tipo de opción especial (aunque el fichero compilado será procesado posteriormente por herramientas específicas para la plataforma Java Card).

No obstante, muchas características del lenguaje Java no son compatibles con Java Card (en particular algunos tipos básicos (char, double, float y long); los enums; los arrays de más de una dimensión; los hilos (threads);... y algunas características son opcionales y están ausentes en la mayoría de las tarjetas inteligentes (pe. el tipo int, en

particular, que es el valor por defecto de un tipo de expresión de Java, la recolección de basura).

En la siguiente tabla se muestran las características soportadas y las no soportadas.

Características Java soportadas	Características Java no soportadas
Tipos primitivos de datos: boolean, byte, short, arrays de una dimensión, paquetes Java, clases, interfaces y excepciones.	Tipos primitivos de datos: long, double, float
Características de orientación a objetos de Java: Herencia, sobrecarga, creación de objetos, ámbitos de acceso.	Caracteres y Strings
El soporte del tipo entero de 32 bits es opcional	Hilos
Recolector de basura	Serialización de objetos, gestión de seguridad

Tabla 2.5: Características del lenguaje Java Card.

- **Hilos**: no se soportan porque las CPUs actuales para tarjetas inteligentes no soportan de manera eficiente la multitarea.
- **Recolector de basura**: todos los objetos que se crean son persistentes, por lo que si consideramos que la memoria de las tarjetas es bastante limitada, ésta es una de las restricciones más importantes a tener en cuenta cuando se realizan aplicaciones Java Card.
- **Arrays:** Sólo se soportan arrays unidimensionales.
- **Tipos primitivos**: Sólo soporta los tipos primitivos mostrados en la Tabla 2.6

Tipo	Tamaño	Rango
bye	8bits	-128,127
short	16bits	-32768, 32767
boolean	8bits	TRUE o FALSE
int	32bits	-2147483648,2147483647

Tabla 2.6 Tipos primitivos soportados por Java Card.

El tipo int no se soporta en todas las plataformas, debido a que la mayoría de CPUs son de 8 ó 16 bits, y por lo tanto el manejo de este tipo de datos es muy costoso. En la especificación de Java Card el tipo int es opcional. No se soportan ni los tipos char, double, flooat o long, ni los modificadores transient y volatile.
Es necesario considerar, que algunas de estas limitaciones pueden no existir en algunos tipos de tarjetas y estar proporcionadas por una extensión específica del fabricante, pero habrá que tener en cuenta que, entonces no se podrán portar estas aplicaciones a otras tarjetas Java Card que no posean estas características.

❖ **Bytecode**

El bytecode de Java Card gestionado por la máquina virtual de Java Card es un subconjunto funcional de Java (Java 2 Standard Edition) aunque utiliza una codificación diferente optimizada para ocupar menos espacio.

❖ **Librerías de ejecución**

Las librerías estándar de Java Card difieren en mucho de las de Java, y el subconjunto común es mínimo. Por ejemplo, la clase del Java Security Manager no es compatible con Java Card, donde las políticas de seguridad son ejecutadas por la máquina virtual de la tarjeta.

❖ **Desarrollo**

Las técnicas de codificación utilizadas en la programación de Java Cards difieren significativamente de las que se utilizan en un programa Java. Aún así, al utilizar Java Card un subconjunto del lenguaje Java se acelera la curva de aprendizaje, y permite utilizar un entorno Java para desarrollar y depurar un programa Java Card.

2.5.3 Java Card API

Además del conjunto de clases dedinido para el lenguaje Java, "Java Card Framework" define un conjunto de clases específico para soportar las aplicaciones Java Card. Estas están contenidas en los siguientes paquetes [31]:

➢ **java.lang**
- o Clase Object Es la raíz de la jerarquía de clases de la Plataforma Java Card.
- o Clase Throwable Es la superclase de todos los errores y excepciones en la plataforma Java Card.

➢ **javacard.framework**

Define los interfaces, clases y excepciones que componen el núcleo de "Java Card Framework". Especifica conceptos importantes tales como la "Unidad de datos del protocolo de aplicación" (APDU), el Java Card applet (Applet), el Java Card System (JCSystem), el número de Identificación Personal (PIN). Define también varias constantes ISO7816 y varias excepciones específiccas de Java Card.

▪ **Interfaces**
- o ISO7816: Define constantes relativas al ISO 7816-3 y ISO 7816-4.
- o MultiSelectable: Identifica applets que pueden soportar selecciones concurrentes.
- o PIN: Representa un número de identificación personal usado con motivos de seguridad (autenticación).
- o Shareable: Identifica un objeto compartido. Los objetos que deben estar disponibles a través del firewall en el ámbito de los applets deben implementar este interfaz.

▪ **Clases**
- o AID: Define un identificador de aplicación, conforme al estándar ISO7816-5. Es un atributo obligatorio para un applet.
- o APDU: Unidad de datos del Protocolo de Aplicación, constituye el formato de comunicación entre el applet en la tarjeta y el terminal móvil.
- o Applet: Aplicación Java Card. Todos los applets deben extender de esta clase abstracta.

- JCSystem: Proporciona métodos de control del ciclo de vida de un applet, gestión de recursos y transacciones y compartición de objetos entre applets así como borrado de objetos.
- OwnerPIN: Es una implementación del interfaz “PIN".
- Util: Proporciona métodos para la manipulación de arrays y shorts, como arrayCompare(), arrayCopy(), arrayCopyNonAtomic(), arrayFillNonAtomic(), getShort(), makeShort(), setShort().

- **Excepciones**

Se especifícan varias clases de excepciones: APDUException, CardException, CardRuntimeException, ISOException, PINException, SystemException, TransactionException, UserException.

- **javacard.security**

- **Interfaces**
- Interfaces de base genérica: Key, PrivateKey, PublicKey, y SecretKey.
- Subinterfaces: AESKey, DESKey, DSAKey, DSAPrivateKey, DSAPublicKey, ECKey, ECPrivateKey, ECPublicKey, RSAPrivateCrtKey, RSAPrivateKey, RSAPublicKey,
- KoreanSEEDKey, HMACKey.
- Representan varios tipos de claves de seguridad y algoritmos.
- SignatureMessageRecovery: Interfaz que será implementada por una subclase de firma para proporcionar la funcionalidad de recuperación de mensajes.

- **Clases**
- Checksum: Clase abstracta base de los algoritmos CRC.
- KeyAgreement: Base de los algoritmos de acuerdos de clave.
- KeyBuilder: Constructor de objetos Clave.
- KeyPair: Contenedor del par de claves: pública y privada.
- MessageDigest: Base para los algoritmos de Hash.
- InitializedMessageDigest: Subclase de MessageDigest para inicializar el valor de la función de Hash. Se usa para calcular una funci_on resumen como subconjunto de un mensaje de datos completo.
- RandomData: Clase que genera números aleatorios.
- Signature: Clase abstracta para algoritmos de firrma.

- **Excepciones**
- CryptoException: Se trata de excepciones relacionadas con la encriptación tales como algoritmos no soportados o claves no inicializadas.

- **javacard.crypto**

- **Interfaces**
- KeyEncryption: Interfaz usado para desencriptar una clave de entrada empleada para algoritmos de encriptación.

- C**lases**
  - Cipher: Clase abstracta que todos los cifradores deben implementar.

2.5.4 JavaCard APPLET

Los applets son las aplicaciones que corren en una Java Card. Dichas aplicaciones interactúan en todo momento con el JCRE utilizando los servicios que éste ofrece, e implementan la interfaz definida en la clase abstracta javacard.framework.Applet, la cual deben extender. [33]

Se puede decir que un applet comienza su ciclo de vida al ser correctamente cargado en la memoria de la tarjeta y preparado para su correcta ejecución. Para ello es necesario realizar los siguientes pasos:

1. Compilar el código fuente para obtener un fichero .CLASS
2. Convertir los ficheros CLASS a imágenes binarias descargables en la tarjeta, ficheros CAP.
3. Descargar los ficheros CAP en la tarjeta inteligente.

Una vez que el applet Java Card se encuentra en la tarjeta, es necesario instalarlo y registrarlo:

1. **Instalación.**

El proceso consiste en crear una instancia del applet que hemos descargado en la tarjeta. Esta instancia se crea en memoria EEPROM, y permanecerá en la tarjeta hasta que sea borrada. Como se ha indicado las tarjetas no poseen recolector de basura, por lo que la gestión del almacenamiento de aplicaciones descargadas en la tarjeta es un tema abierto.

2. **Registro.**

Cada applet que se instala en una tarjeta debe ser registrado en el JCRE con un AID único. El formato de estos identificadores está estandarizado en el ISO 7816/5.

Un AID (Aplication Identifier) es uncadena de entre 5 y 16 bytes en hexadecimal.

- Los 5 primeros bytes del AID identifican únicamente al proveedor del applet que es la compañía que proporciona el paquete o applet. Este es el denominado Resource Identifier (RID). El proveedor debe registrar el RID en la ISO para que pueda ser empleado correctamente.
- Los 11 bytes restantes del AID contienen el Propietary Identifier Extension (PIX). El PIX identifica unequívocamente al paquete, applet o instancia de applet. Este identificador es asignado por el proveedor.

En todos los casos los bytes 13, 14 y 15 del PIX están reservados para la Referencia de Aplicación Toolkit (TAR). El TAR es un código de 3 bytes utilizado para identificar unequívocamente a un segundo nivel de aplicación (por ejemplo una aplicación Toolkit).

Una vez instalada y registrada, para que un applet comience a ejecutarse debe de ser seleccionada a través de su identificador AID, esta selección se realiza enviando a la tarjeta una APDU SELECT definido en el estándar ISO 7816 parte 4. En un momento determinado, sólo puede existir un applet ejecutándose en una tarjeta, el JCRE es el

encargado de realizar las tareas de gestión de la ejecución de las aplicaciones instaladas en las tarjetas.

2.5.5 APDU

En los procesos de comunicación se intercambian paquetes de datos, los cuales son construidos siguiendo un conjunto de protocolos.

En forma similar, las tarjetas inteligentes se comunican al mundo exterior usando sus propios paquetes de datos llamados APDU (Application Protocol Data Units). Los APDU pueden contener un mensaje de comando o un mensaje de respuesta. En el mundo de las tarjetas, el modelo maestro – esclavo es el más usado, por lo cual una tarjeta inteligente siempre juega el rol pasivo. En otras palabras, una tarjeta inteligente siempre espera un comando APDU. Cuando lo recibe, ejecuta la acción especificada en el APDU y responde con un APDU respuesta. APDUs comandos y APDUs respuestas son intercambiados alternativamente. [34]

La tabla 2.7 ilustra el formato de un APDU comando y un APDU respuesta, respectivamente. La estructura APDU se describe en la parte 4 del ISO 7816.

APDU Comando						
Encabezado Obligatorio				Cuerpo Opcional		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**
APDU Respuesta						
Cuerpo Opcional	Cola Obligatoria					
Data field	**SW1**		**SW2**			

Tabla 2.7: Estructura APDU

El encabezamiento codifica el comando seleccionado. Este consiste de cuatro campos: clase (**CLA**), instrucción (**INS**), y parámetros 1 y 2 (**P1** y **P2**). Cada campo contiene 1 byte:

- **CLA**: byte Clase. En muchas tarjetas inteligentes, este byte es usado para identificar una aplicación.
- **INS**: byte Instrucción. Este byte indica el código de la instrucción.
- **P1**-**P2**: bytes Parámetros. Estos proporcionan más opciones al APDU comando.

Lc denota el número de bytes dentro del **Data field** del APDU comando; **Le** denota el máximo número de bytes esperados en el **Data field** del siguiente APDU respuesta.

Los bytes **SW1** y **SW2** denotan el estado de la respuesta del comando APDU en una tarjeta. Si es 90 00 indica que la operación solocitada se ha realizado con éxito.

2.6 Servidor Web TOMCAT

Un servidor web es un programa que se ejecuta continuamente en un ordenador, manteniéndose a la espera de peticiones que le hará un cliente. El servidor web se encarga de contestar a estas peticiones de forma adecuada, entregando como resultado una página web o información de todo tipo de acuerdo a los comandos solicitados. [35]

Tomcat (también llamado Jakarta Tomcat o Apache Tomcat) funciona como un contenedor de servlets desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de JavaServer Pages (JSP) de Sun Microsystems.Tomcat es un servidor web con soporte de servlets y JSPs, no es un servidor de aplicaciones. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta con el servidor web Apache.

Figura 2.23: Logo del servidor web Tomcat.

Tomcat puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad.

Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java. Es mantenido y desarrollado por miembros de la Apache Software Foundation y voluntarios independientes. Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la Apache Software Licence. Las primeras distribuciones de Tomcat fueron las versiones 3.0.x. Las versiones más recientes son las 6.x, que implementan las especificaciones de Servlet 2.5 y de JSP 2.1. A partir de la versión 4.0, Jakarta Tomcat utiliza el contenedor de servlets Catalina.

2.6.1 Versiones

- **Tomcat 3.x** (distribución inicial)

  - Implementado a partir de las especificaciones Servlet 2.2 y JSP 1.1.
  - Recarga de servlets.
  - Funciones básicas HTTP.

- **Tomcat 4.x**

  - Implementado a partir de las especificaciones Servlet 2.3 y JSP 1.2.
  - Contenedor de servlets rediseñado como Catalina.
  - Motor JSP rediseñado con Jasper.
  - Conector Coyote.
  - Java Management Extensions, JSP y administración basada en Struts.

- **Tomcat 5.x**

  - Implementado a partir de las especificaciones Servlet 2.4 y JSP 2.0.
  - Recolección de basura reducida.
  - Capa envolvente nativa para Windows y Unix para la integración de las plataformas.
  - Análisis rápido JSP.

- **Tomcat 6.x (Utilizado en el proyecto)**

  - Implementado de Servlet 2.5 y JSP 2.1.
  - Soporte para *Unified Expression Language* 2.1.
  - Diseñado para funcionar en Java SE 5.0 y posteriores.
  - Soporte para Comet a través de la interfaz CometProcessor.

- **Tomcat 7.x**

  - Implementado de Servlet 3.0 JSP 2.2 y EL 2.2.
  - Mejoras para detectar y prevenir "fugas de memoria" en las aplicaciones web.
  - Limpieza interna de código.
  - Soporte para la inclusión de contenidos externos directamente en una aplicación web.

2.7 GESTIÓN DE LA BASE DE DATOS

Los sistemas de gestión de bases de datos (en inglés *database management system*, abreviado *DBMS*) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

Existen distintos objetivos que deben cumplir los SGBD:

- **Abstracción de la información**. Ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo cuanto ocupe la base de datos, esto es transparente al usuario. Así, se definen varios niveles de abstracción.
- **Independencia**. La independencia de los datos consiste en la capacidad de modificar el esquema de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.
- **Consistencia**. En aquellos casos en los que no se ha logrado eliminar la redundancia, será necesario vigilar que todos los datos repetidos se actualicen de forma simultánea. Por otra parte, la base de datos representa una realidad determinada que tiene determinadas condiciones, por ejemplo que los menores de edad no pueden tener licencia de conducir. El sistema no debería aceptar datos de un conductor menor de edad. En los SGBD existen herramientas que facilitan la programación de este tipo de condiciones.
- **Seguridad**. La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra segura de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos.
- **Manejo de transacciones**. Una transacción es un programa que se ejecuta como una sola operación. Esto quiere decir que luego de una ejecución en la que se produce una falla es el mismo que se obtendría si el programa no se hubiera ejecutado. Los SGBD proveen mecanismos para programar las modificaciones de los datos de una forma mucho más simple que si no se dispusiera de ellos.
- **Tiempo de respuesta**. Lógicamente, es deseable minimizar el tiempo que el SGBD tarda en darnos la información solicitada y en almacenar los cambios.

2.7.1 APACHE DERBY

Apache Derby es un sistema gestor de base de datos relacional escrito en Java que puede ser añadido en aplicaciones Java y utilizado para procesos de transacciones online. Tiene un tamaño de 2 MB de espacio en disco. Inicialmente distribuido como IBM Cloudscape, Apache Derby es un proyecto *open source* licenciado bajo la Apache 2.0 License. Actualmente se distribuye como Sun Java DB. [36]

Figura 2.24: Logo del gestor de base de datos relacional Derby.

2.7.1.1 Características

- APIs para JDBC y SQL. Soporta todas las características de SQL92 y la mayoría de SQL99. La sintaxis SQL usada proviene de IBM DB2.
- Su código mide alrededor de 2000KB comprimido.
- Soporta cifrado completo, roles y permisos. Además posee SQL SCHEMAS para separar la información en un única base de datos y control completo de usuarios.
- Soporta internamente procedures, cifrado y compresión.
- Trae soporte multilenguaje y localizaciones específicas.
- A partir de la versión 10.4 trae un sistema simple de replicación maestro-esclavo.
- Transacciones y recuperación ante errores ACID.
- Posee tres productos asociados a la marca:
    - Derby Embedded Database Engine: El motor propiamente dicho.
    - Derby Network Server: Permite convertir Derby en una base de datos que sigue el modelo cliente-servidor tradicional.
    - Database Utilities: Un paquete de utilidades.

Derby está escrito en Java y no tiene *bindings* para otros lenguajes por lo que limita al programador a utilizarlo mediante la máquina virtual de Java y en programas escritos en ese lenguaje o lenguajes de scripting que se ejecuten sobre JVM (Jython, JRuby, Jacl, etc.). Esto por otro lado hace que las aplicaciones sean altamente portables.

En su modo empotrado sólo soporta un único proceso que tenga abierta la base de datos. Sin embargo en su modo de cliente/servidor soporta el acceso de varios procesos simultáneos mediante bloqueo de filas.

2.8 Servlet

Los servlets, son objetos que corren dentro del contexto de un contenedor de servlets, como Tomcat, y extienden su funcionalidad.El uso más común de los servlets es generar páginas web de forma dinámica a partir de los parámetros de la petición que envíe el navegador web. [37]

Un servlet es un objeto que se ejecuta en un servidor o contenedor JEE, especialmente diseñado para ofrecer contenido dinámico desde un servidor web, generalmente HTML. Otras opciones que permiten generar contenido dinámico son los lenguajes ASP, PHP, JSP (un caso especial de servlet), Ruby y Python. Forman parte de JEE (Java Enterprise Edition), que es una ampliación de JSE (Java Standard Edition).

Un servlet implementa la interfaz javax.servlet.Servlet o hereda alguna de las clases más convenientes para un protocolo específico (ej: javax.servlet.HttpServlet). Al implementar esta interfaz el servlet es capaz de interpretar los objetos de tipo HttpServletRequest y HttpServletResponse quienes contienen la información de la página que invocó al servlet.

Entre el servidor de aplicaciones y el servlet existe un contrato que determina cómo han de interactuar. La especificación de éste se encuentra en los JSR (Java Specification Requests) del JCP (Java Community Process).

2.8.1 Ciclo de Vida

El ciclo de vida de un Servlet se divide en los siguientes puntos:

1. El cliente solicita una petición a un servidor vía URL.
2. El servidor recibe la petición.
    1. Si es la primera, se utiliza el motor de Servlets para cargarlo y se llama al método init().
    2. Si ya está iniciado, cualquier petición se convierte en un nuevo hilo. Un Servlet puede manejar múltiples peticiones de clientes.
3. Se llama al método service() para procesar la petición devolviendo el resultado al cliente.
4. Cuando se apaga el motor de un Servlet se llama al método destroy(), que lo destruye y libera los recursos abiertos.

2.9 JSP

JavaServer Pages (**JSP**) es una tecnología Java que permite generar contenido dinámico para web, en forma de documentos HTML, XML o de otro tipo. [38]

Esta tecnología es un desarrollo de la compañía Sun Microsystems. La Especificación JSP 1.2 fue la primera que se liberó y en la actualidad está disponible la Especificación JSP 2.1.

Las JSP's permiten la utilización de código Java mediante scripts. Además, es posible utilizar algunas acciones JSP predefinidas mediante etiquetas. Estas etiquetas pueden ser enriquecidas mediante la utilización de Bibliotecas de Etiquetas (TagLibs) externas e incluso personalizadas.

2.9.1 Arquitectura

JSP puede considerarse como una manera alternativa, y simplificada, de construir servlets. Es por ello que una página JSP puede hacer todo lo que un servlet puede hacer, y viceversa. Cada versión de la especificación de JSP está fuertemente vinculada a una versión en particular de la especificación de servlets.

El funcionamiento general de la tecnología JSP es que el Servidor de Aplicaciones interpreta el código contenido en la página JSP para construir el código Java del servlet a generar. Este servlet será el que genere el documento (típicamente HTML) que se presentará en la pantalla del Navegador del usuario.

JSP -> Servidor Aplicaciones (Servlets) -> Cliente (Navegador)

Es posible enriquecer el lenguaje de etiquetas utilizado por JSP. Para ello debemos extender la capa de alto nivel JSP mediante la implementación de Bibliotecas de Etiquetas (Tags Libraries). Un ejemplo de estas bibliotecas son las propocionadas por Sun bajo la denominación de JSTL o las distribuidas por Apache junto con el Framework de Struts.

TagLibs -> JSP -> Servidor Aplicaciones (Servlets) -> Cliente (Navegador)

El rendimiento de una página JSP es el mismo que tendría el servidor equivalente, ya que el código es compilado como cualquier otra clase Java. A su vez, la máquina virtual compilará dinámicamente a código de máquina las partes de la aplicación que lo requieran. Esto hace que JSP tenga un buen desempeño y sea más eficiente que otras tecnologías web que ejecutan el código de una manera puramente interpretada.

2.9.2 Ventajas JSP’s

La principal ventaja de JSP frente a otros lenguajes es que el lenguaje Java es un lenguaje de propósito general que excede el mundo web y que es apto para crear

clases que manejen lógica de negocio y acceso a datos de una manera prolija. Esto permite separar en niveles las aplicaciones web, dejando la parte encargada de generar el documento HTML en el archivo JSP.

Otra ventaja es que JSP hereda la portabilidad de Java, y es posible ejecutar las aplicaciones en múltiples plataformas sin cambios. Es común incluso que los desarrolladores trabajen en una plataforma y que la aplicación termine siendo ejecutada en otra.

2.9.3 JSP's y Servlets

Los servlets y Java Server Pages (JSPs) son dos métodos de creación de páginas web dinámicas en servidor usando el lenguaje Java. En ese sentido son similares a otros métodos o lenguajes tales como el PHP, ASP o los CGIs, programas que generan páginas web en el servidor. Sin embargo, se diferencian de ellos en otras cosas.

Para empezar, los JSPs y servlets se ejecutan en una máquina virtual Java, lo cual permite que, en principio, se puedan usar en cualquier tipo de ordenador, siempre que exista una máquina virtual Java para él. Cada servlet (o JSP, a partir de ahora lo usaremos de forma indistinta) se ejecuta en su propia hebra, es decir, en su propio contexto; pero no se comienza a ejecutar cada vez que recibe una petición, sino que persiste de una petición a la siguiente, de forma que no se pierde tiempo en invocarlo (cargar programa + intérprete). Su persistencia le permite también hacer una serie de cosas de forma más eficiente: conexión a bases de datos y manejo de sesiones, por ejemplo.

Los JSPs son en realidad servlets: un JSP se compila a un programa en Java la primera vez que se invoca, y del programa en Java se crea una clase que se empieza a ejecutar en el servidor como un servlet. La principal diferencia entre los servlets y los JSPs es el enfoque de la programación: un JSP es una página Web con etiquetas especiales y código Java incrustado, mientras que un servlet es un programa Java puro que recibe peticiones y genera a partir de ellas una página web.

Capítulo 3

Visión de alto nivel de la aplicación

La aplicación desarrollada en el proyecto está pensada para obtener información en cualquier lugar de la universidad (despachos, laboratorios, aulas, bibliotecas…) que esté dotado de un lector NFC y sea accesible para cualquier usuario que posea un móvil con la tecnología NFC (por ejemplo un Nokia 6131 o Nokia 6212). El usuario tan sólo necesitará tener en el elemento seguro de su móvil su identificador (NIA) y una clave que le será solicitada cuando consulte cierta información restringida (reserva de libros, solicitación de tutorías…). En caso de que no disponga del móvil siempre podrá obtener la misma funcionalidad desde la aplicación web.

3.1 Requisitos

Para llevar a cabo este proyecto se han requerido varios elementos de hardware y software que se describen a continuación:

3.1.1 Software

- La plataforma en la que se ha desarrollado el proyecto es **Netbeans 6.9.**

- El proyecto se ha realizado con el lenguaje de programación **Java**, versión **1.6.3** para la aplicación y versión **1.3** para la creación de applets.
- Para simular el proyecto se ha utilizado el **NOKIA NFC SDK** intregado en Netbeans.
- Para grabar el applet en el elemento seguro o borrarlo, se ha utilizado **GPSHELL** (GlobalPlatformShell), versión **1.4.2** GlobalPlatform es un estándar para gestionar el contenido de una smartcard, es este caso, el elemento seguro del móvil. Principalmente se encarga de añadir y eliminar aplicaciones.
- **JavaCardKit,** que se encarga de generar los arhivos .CAP que son los que se graban en el elemento seguro a partir de los archivos .CLASS generardos al compilar el applet con la versión de Java 1.3.La versión utilizada de JavaCardKit es la **2.2.1** ya que al principio se probó con la versión mas nueva, JavaCardKitConnected 3.0 pero el archivo .CAP generado con ésta no era compatible con el elemento seguro del móvil.
- **Tomcat,** que es el servidor web empleado, con la versión **6.0**
- **Derby,** que es el sistema de gestión de la base de datos empleado.

Figura 3.1: Requisitos Software

3.1.2 Hardware

- Ordenador con los requisistos software mencionados en el punto anterior.
- Dos móviles Nokia 6131, ya que poseen la tecnología NFC y Bluetooth. Uno actúa como lector y el otro es el que contiene la aplicación.
- Dos móviles Nokia 6212, ya que además de las características que soportan los Nokia 6131 permite enviar datos cifrados MD5 mediante NFCIP.
- Omnikey Reader 5321 que es el elemento a través del cual se pueden grabar los applets en el elemento seguro del móvil a través de la GPShell.

Figura 3.2 Requisitos Hardware

3.2 Interacción entre componentes

3.2.1 Componentes aplicación móvil

En la aplicación móvil, los componentes principales que intervienen son:

- **Ordenador:** Contiene los requisitos software descritos en el apartado 3.1.1. Con el ordenador se pueden crear los applets y grabarlos en el Elemento Seguro del móvil real a través de GPShell. También permite la creación y simulación del proyecto y de solicitar al Omnikey la autenticación cuando sea necesario durante la simulación.

- **Omnikey Reader 5321:** Permite la comunicación entre el ordenador y el elemento seguro del móvil. Es el intermediario entre las comunicaciones que se realizan entre ambos: el ordenador puede grabar applets en el elemento seguro del móvil a través de él y también solicitarle la autenticación del usuario, para lo cual el Omnikey accede al elemento seguro del móvil y cuando las obtiene se las envía al ordenador.

- **Móvil:** Como ya se ha mencionado anteriormente, el modelo del móvil puede ser un Nokia 6131 NFC o un 6212 NFC. El móvil es el que contiene el applet en el elemento seguro. Para poder instalar applets en el elemento seguro es necesario haber desbloquedao previamente el elemento seguro del móvil, como ya se comentará posteriormente en esta memoria.

Figura 3.3 Esquema de interacción en la aplición móvil

Como se puede ver en la Figura 3.3, cuando se está simulando la aplicación móvil en NetBeans, cuando se desea acceder a algún recurso que requiere autenticación, el simulador se la solicita al Lector Omnikey (1). Para obtenerla, el lector Omnikey le manda un comando APDU al Elemento Seguro del móvil (2), seleccionando el Applet y solicitándole la autenticación. Al recibirlo, el Applet cifra la contraseña y se la envía al Ominkey junto con el NIA para identificar al alumno (3). Por último, el Omnikey le envía los datos recibidos al simulador (4), y este se encarga de comprobar si la contraseña es correcta y si el usuario tiene los permisos para acceder al recurso solicitado.

3.2.2 Componentes aplicación web

- **Ordenador:** Es el único elemento hardware necesario para poder ejecutar la aplicación Web es el ordenador. El ordenador interactúa con la aplicación web, la autenticación es requerida al incio de la aplicación y si no es correcta no se podrá ejecutar la aplicación web.

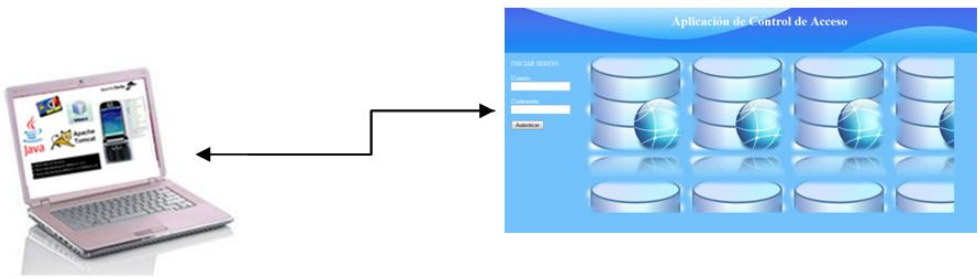


Figura 3.4 Esquema de interacción en la aplición web

3.3 Ejecución de la aplicación

3.3.1 Aplicación móvil

3.3.1.1 Simulación de NFC

El usuario sólo ha de llevar consigo su móvil, con la autenticación del elemento seguro del móvil será suficiente. La autenticación se lleva a cabo cuando el MIDlet solicita al elemento seguro la contraseña y éste se la devuelve cifrada en MD5. El MIDlet la envía al servidor y éste comprueba si es correcta. El acceso al elemento seguro podría hacerse mediante la simulación de una SmartCard a través de una clase que extendiera de *InternalSecureCard* o bien, se podría simular que siempre devolviera un ACK. En este proyecto se hace de forma real a través del Omnikey Reader que está conectado con el PC y con el móvil. El simulador accede al Applet del elemento seguro del Nokia 6131.

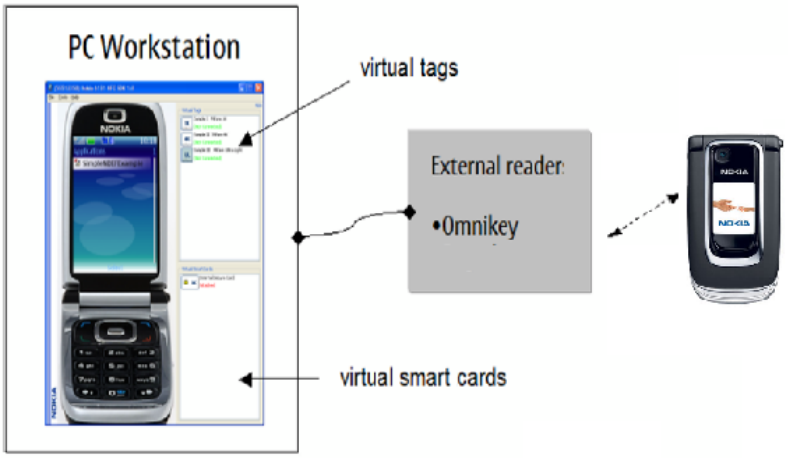


Figura 3.5 Acceso al elemento seguro a través del Omnikey

Dependiendo de la ubicación en la que se encuentre obtendrá distinta información:

- **Despacho:** Se pordrá consultar quién es el profesor que posee ese despacho o solicitar una tutoría. Si elije esta opción, deberá autenticarse y tras esto, le aparecerá en el móvil una lista con las fechas disponibles y tras seleccionar una le aparecerá las horas que tiene disponibles en esa fecha y tras elegirla, la tutoría queda ya fijada. Posteriormente, cuando acuda el profesor, al tocar con su móvil y autenticarse con su contraseña obtendrá una lista con las tutorías fijadas (con fecha y hora) y el alumno que la ha solicitado.
- **Sala de Ordenadores**: Se podrán consultar las características de los ordenadores de esa Sala y cuántos hay libres y cuántos ocupados y, tras la autenticación, se podrá reservar un ordenador o consultar el horario de la Sala
- **Aula:** Se podrá consultar los profesores que van a dar clase y el departamento al que pertenecen y, tras la autenticación, se podrá conocer el horario y qué asignaturas se impartirán ó el grupo, profesor y la hora a la que será la clase.

- **PIC:** Para este recurso no se necesita autenticación, está abierto a todo el mundo por tratarse de un sitio de información. Se puede consultar su horario, su localización o su contacto (email y teléfono). Existen tres lectores PIC diferentes: el del campus de Leganés, el de Getafe y el de Colmenarejo.
- **Cafeteria:** Se puede consultar de que se compone el menú del día y los distintos platos combinados pero para reservarlo es necesaria la autenticación.
- **Laboratorio:** Sin la autenticación se puede consultar qué profesores darán clase y a qué departamento pertenecen. Tras autenticarse, se podrá conocer el horario del grupo y el profesor que impartirá la clase ó el horario de la asignatura y el profesor que la impartirá.
- **Libro:** Se puede consultar la información del libro y si el usuario se autentica, podrá conocer para qué asignaturas está recomendado y si lo desea, reservarlo.

Figura 3.6: Posibles ubicaciones que se pueden consultar

3.3.1.2 Bluetooth y NFCIP

Cuando un usuario desea enviar a un lector la reserva realizada, primero tiene que autenticarse para poder acceder a la base de datos y obtener el listado de sus reservas. Cuando selecciona la reserva a enviar, tiene la opción de utilizar Bluetooth o NFCIP [39]. Esto se diseño así para dar mayor soporte a la aplicación. La tecnología que se utilice dependerá en muchos casos de la elección del usuario, pero también hay que tener en cuenta que puede que algún lector no soporte alguna de estas tecnologías y, por tanto, haya que enviarla a través de la tecnología disponible.

- **NFCIP**

Si escoge hacerlo mediante NFCIP, el usuario ha de introducir un PIN que ha de ser el mismo que el introducido en el lector, para añadir mayor seguridad. Si el móvil es un Nokia 6212, lo datos se podrán mandar cifrados, si es un Nokia 6131 no lo soporta.

- **Bluetooth**

Si escoge hacerlo mediante Bluetooth, también se habrá de insertar un PIN en los dos extremos y además, los datos se mandan encriptados.

Figura 3.7: Figura que representa la comunicación NFCIP y Bluetooth

3.3.2 Aplicación Web

La aplicación web permite realizar las mismas acciones que las que se llevan a cabo con los móviles. Añade algunas funcionalidades para los profesores, que pueden añadir o eliminar tutorías y pueden consultar las estadísticas de los recursos más visitados por los alumnos. Para los alumnos ofrece las mismas ventajas que la aplicación móvil.

Figura 3.8. Página principal de la Aplicación Web

Tras autenticarse, muestra distintas opciones dependiendo si es profesor o alumno:

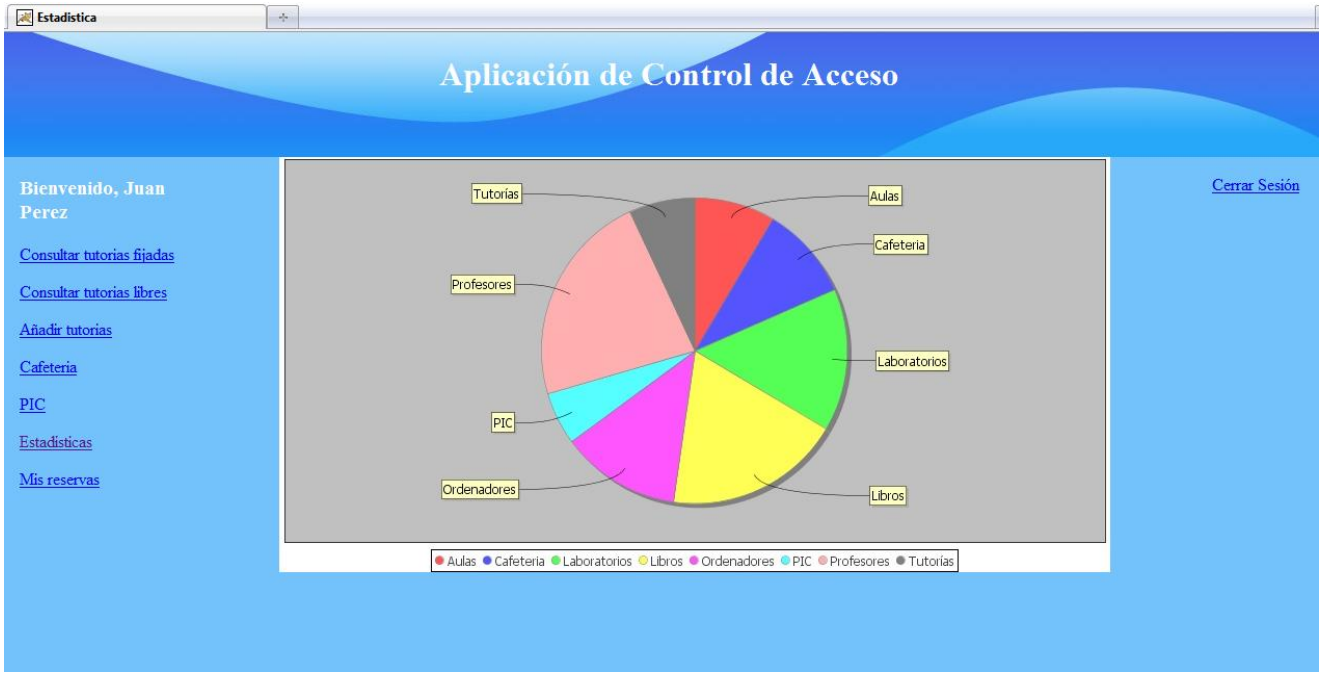


Figura 3.9. Vista de la aplicación desde el punto de vista de un profesor.

Figura 3.10. Vista de la aplicación desde el punto de vista de un alumno.

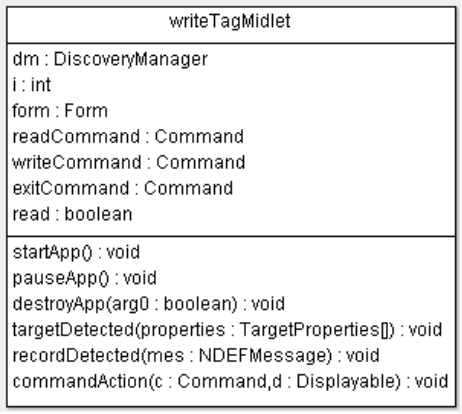
Capítulo 4

Descripción de la Aplicación

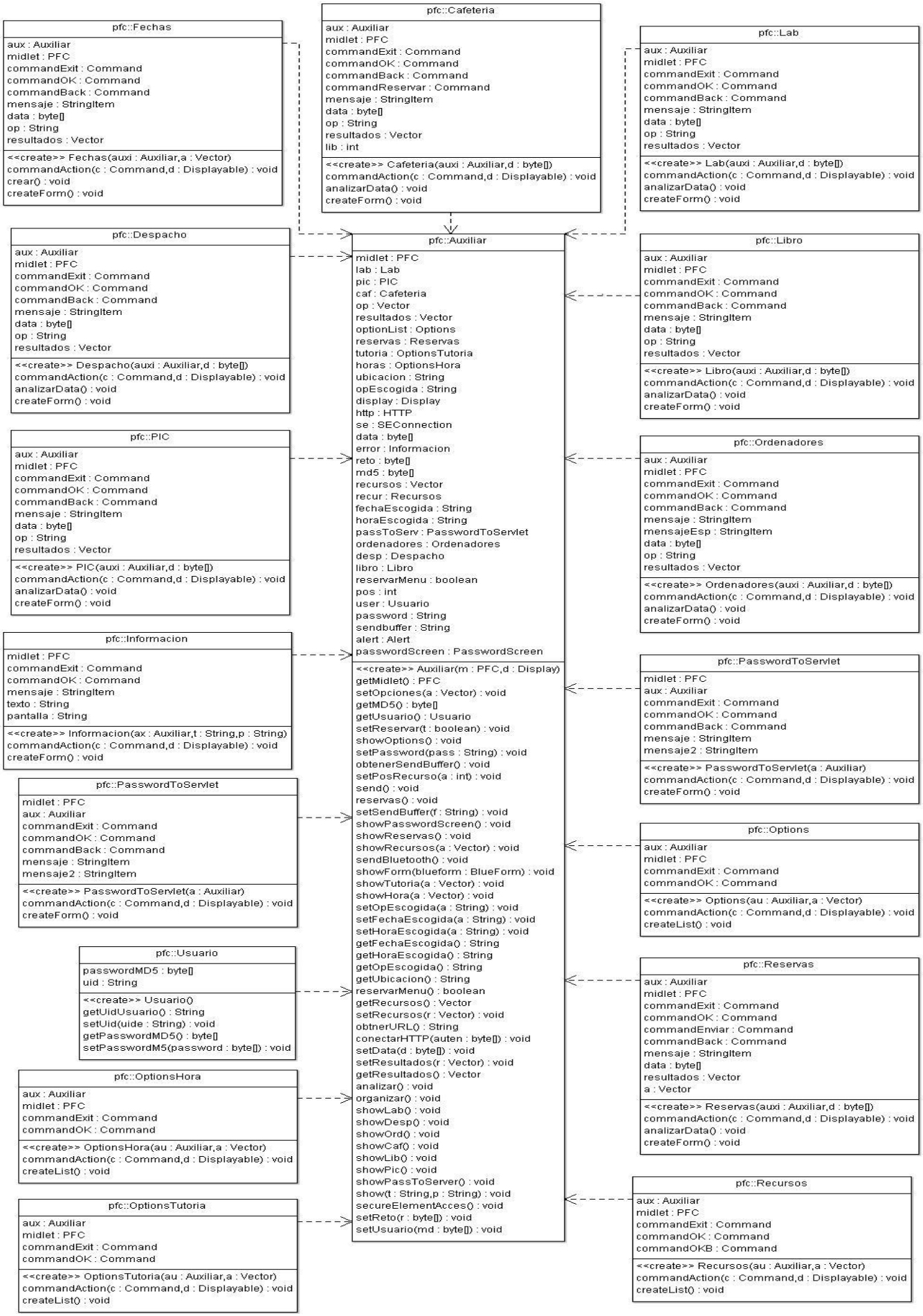
A lo largo de la memoria, se han explicado las tecnologías utilizadas, los requisitos software y hardware requeridos por la aplicación y este capítulo es el encargado de explicar los distintos proyectos que se han desarrollado, asi como su clases y principales métodos.

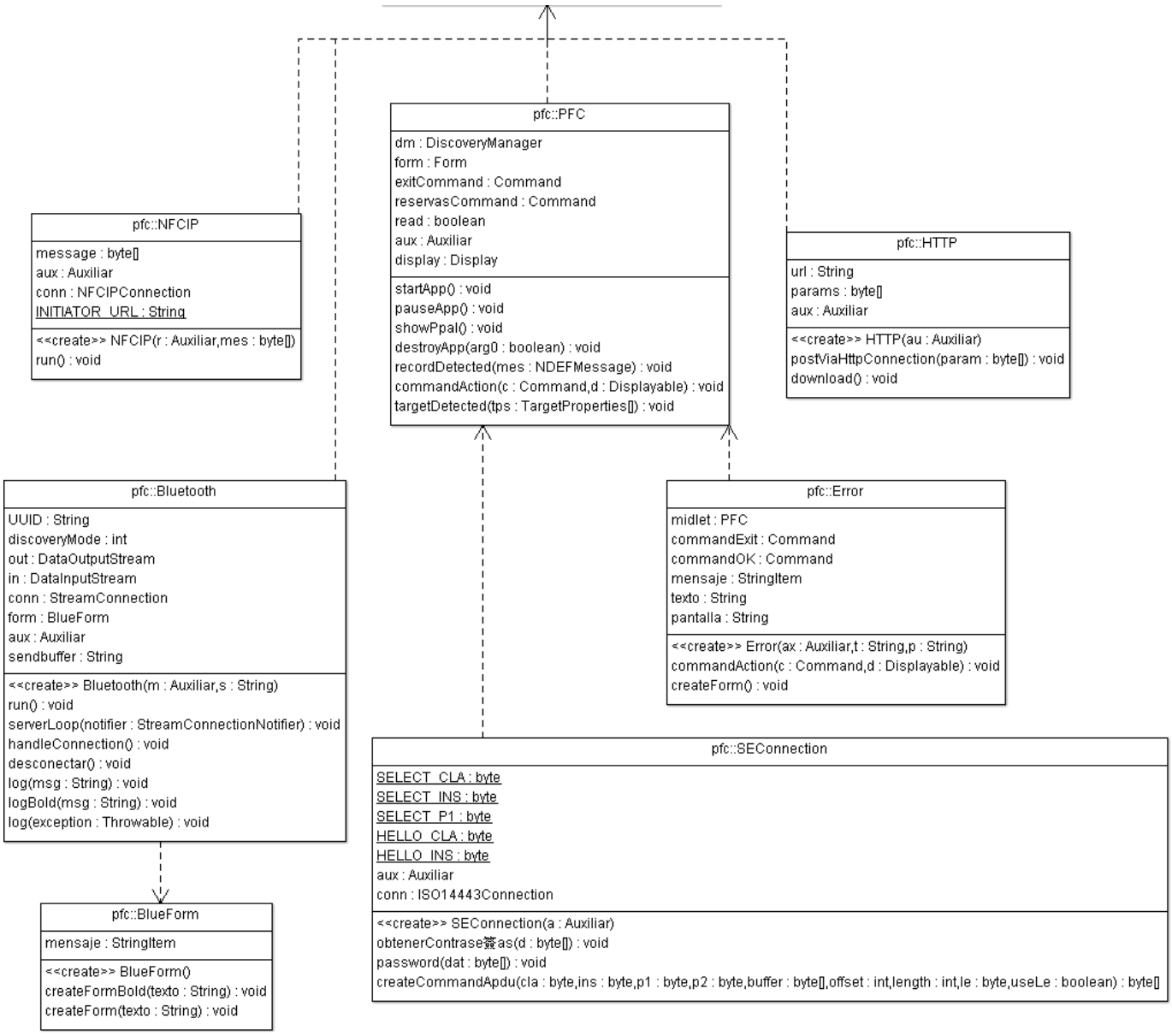
4.1 Diagramas UML

- **Proyecto WriteTags**

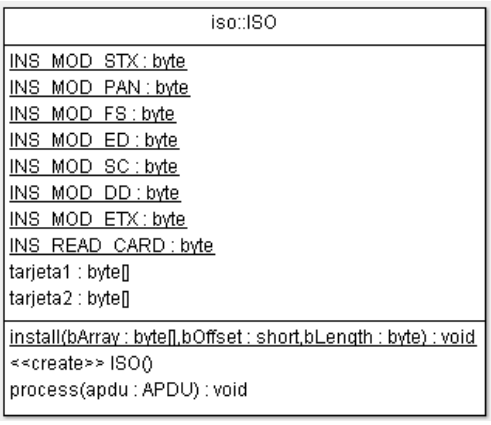


❖ Proyecto PFC

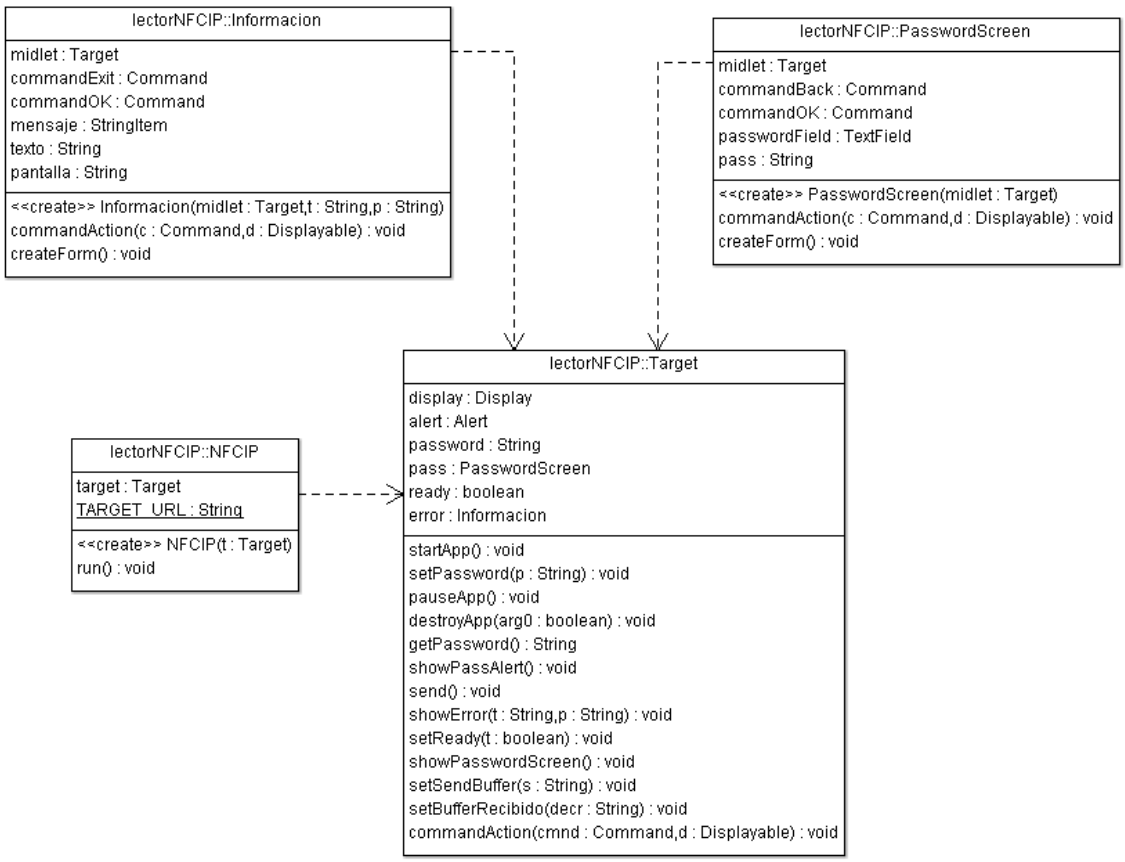




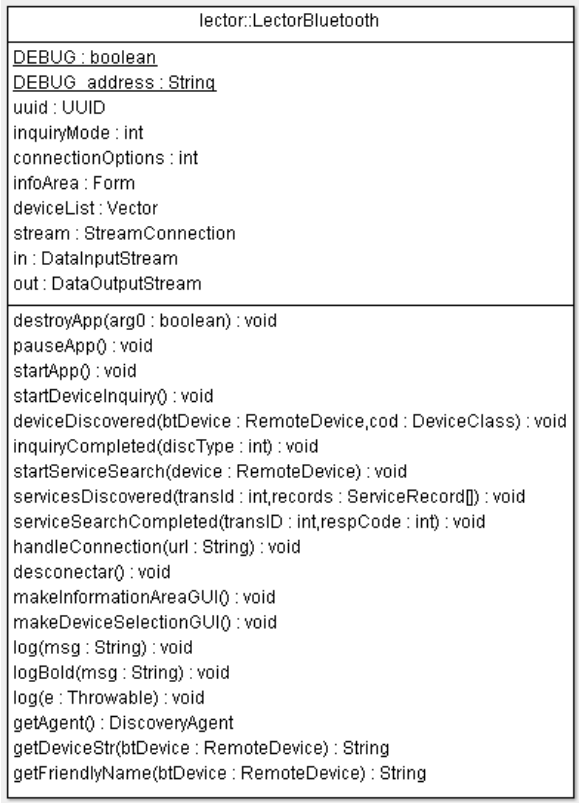
❖ Proyecto SEApplet



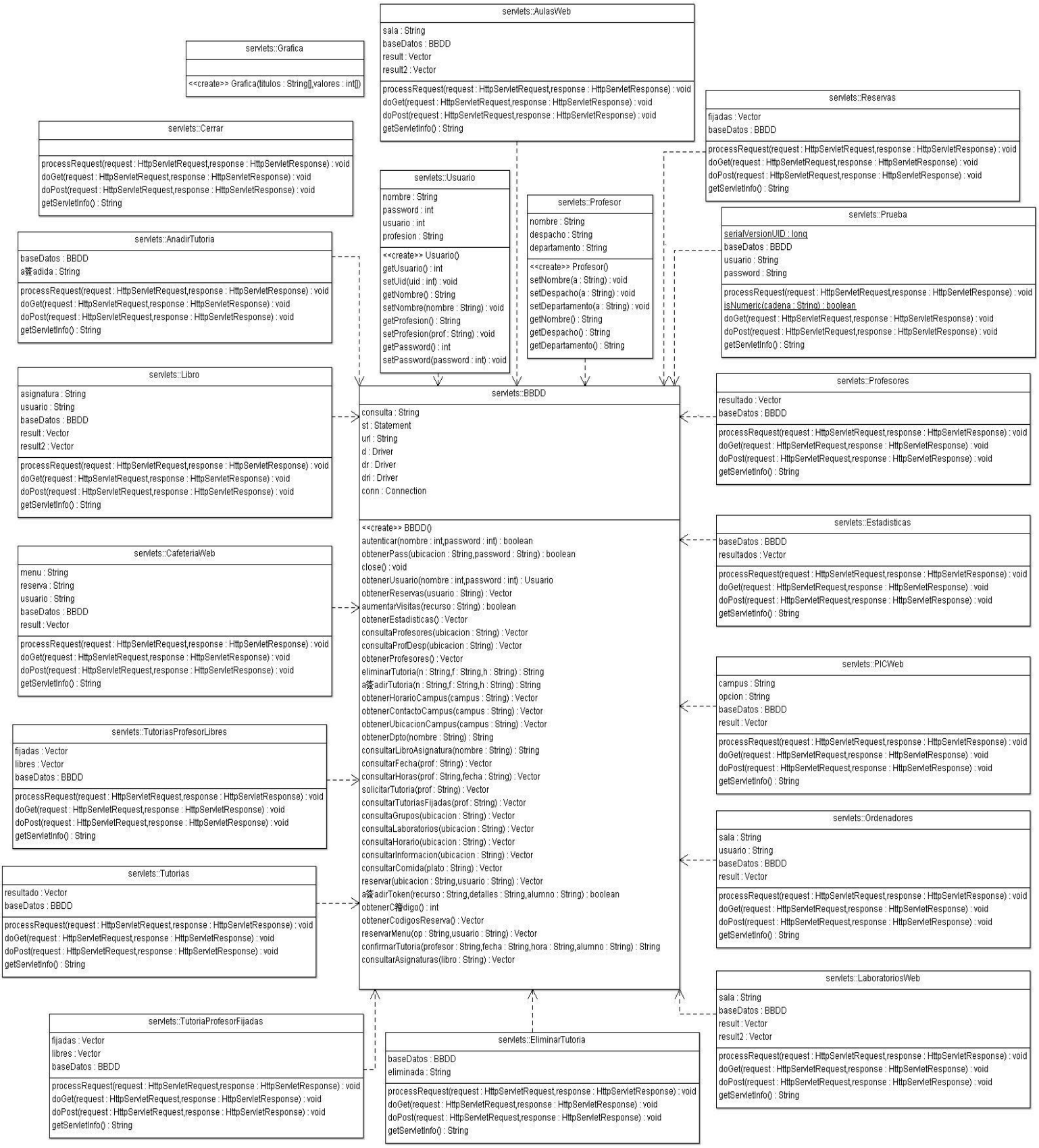
- **Proyecto *LectorNFCIP***



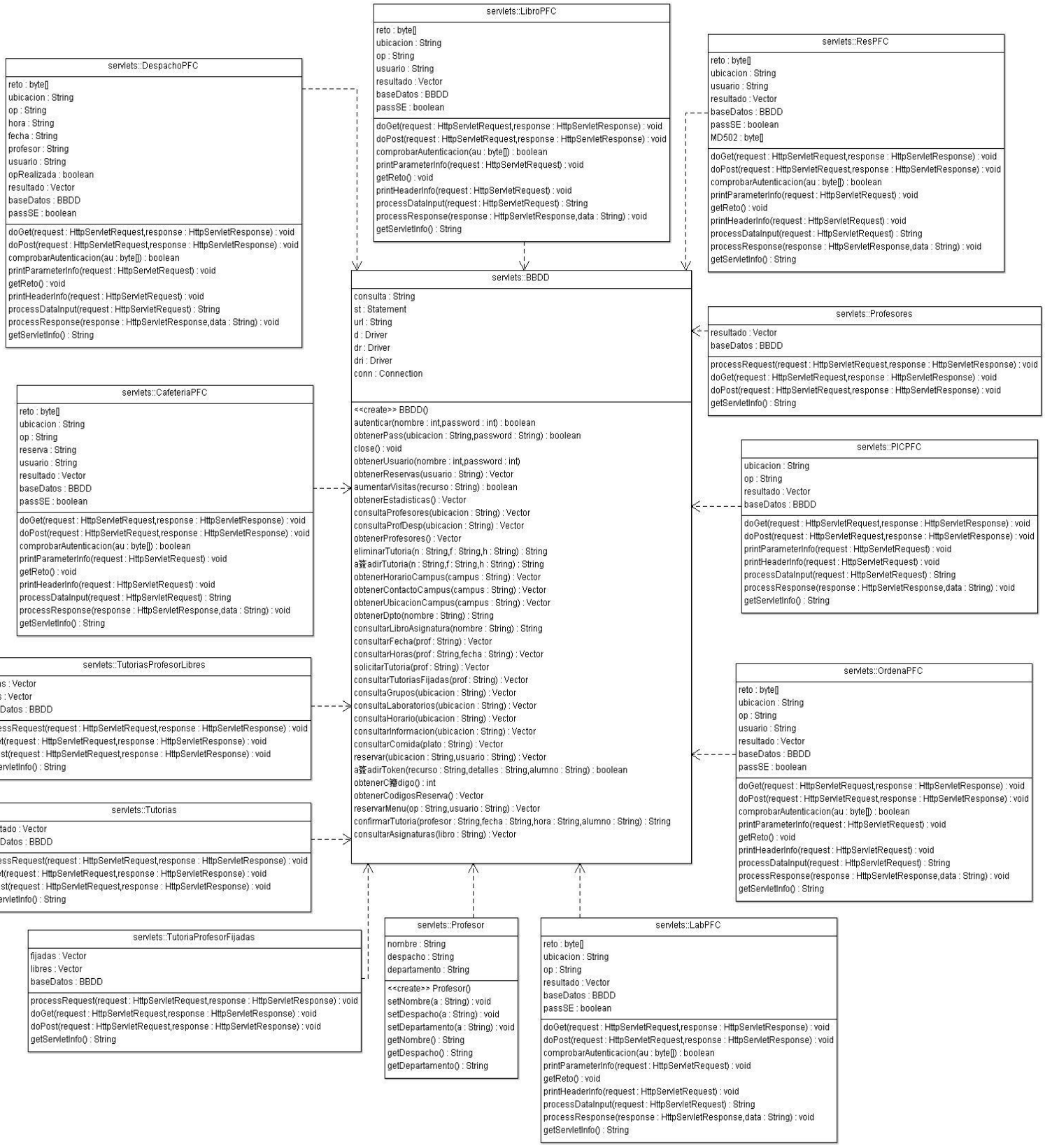
- **Proyecto *LectorBluetooth***



- **Proyecto *ServletsPFC***
  - Servlets de la Aplicación Web



- Servlets de la Aplicación Móvil



4.2 Proyectos y clases desarrolladas

Tras ver la relación existente entre las clases y métodos en apartado anterior, es importante explicar en qué consiste cada clase y sus principales métodos.

4.2.1 Aplicación móvil

La aplicación que se encuentra en el teléfono móvil, la que se encarga de leer las etiquetas en las distintas ubicaciones de la universidad, de mostrar la información solicitada, realizar reservas y enviarlas posteriormente mediante NFCIP o Bluetooth está desarrollada dentro de un proyecto denominado *PFC*.

El proyecto *WriteTag* es el encargado de grabar en los distintos Tags que se encuentran en la universidad el nombre de la unicación y las posibles opciones. Esto será posteriormente leído por la aplicación desarrollada en el proyecto *PFC.*

La aplicación móvil que actúa como lector Bluetooth se encuentra en el proyecto *LectorBluetooth* y la que actúa como lector NFCIP en el proyecto *LectorNFCIP*

La aplicación web está desarrollada a partir del proyecto *ServletsProyecto.* En este proyecto también se encuentran algunos servlets que sirven para gestionar las peticiones HTTP realizadas desde el proyecto *PFC.*

- **Proyecto *WriteTags***
  - **WriteTagMidlet:** Midlet que al acercar un tag graba en él la ubicación y las posibles opciones que se pueden consultar en esa ubicación. Primero ha de crearse un nuevo Tag en el emulador, posteriormente se graba con el proyecto WriteTags y por último hay que seleccionar el tag y seleccionar “Save Data” para que se quede almacenado en el emulador. Al ejecutarlo obtenemos:

Figura 4.1 Proceso de escritura de los Tags

- **Proyecto *SEApplet***

  - **ISO:** Es el Applet que se graba en el elemento seguro y que contiene las respuestas a los posibles comandos recibidos. Contiene la identificación del usuario y la contraseña y cuando el MIDlet se los solicita devuelve el usuario y la contraseña, pero esta última la devuelve cifrada, nunca en claro.

- **Proyecto *PFC***

  Contiene la aplicación principal del proyecto y tiene las siguientes clases:

  - **PFC:** Midlet principal que lee los distintos tags y muestra sus opciones o permite enviar las reservar realizadas al lector mediante NFCIP o Bluetooth.
  - **Bluetooth:** Clase que sirve de gestionar la conexión Bluetooth al enviar reservas.
  - **BlueForm:** Clase que sirve para mostrar el estado de la conexión Bluetooth.
  - **NFCIP:** Clase que gestiona la comunicación NFCIP.
  - **PasswordScreen:** Clase que muestra una pantalla para introducir una contraseña antes de enviar la reserva mediante NFCIP.
  - **HTTP:** Clase que gestiona las conexiones HTTP [40] con el servidor y sus respuestas. Las peticiones se dirigen a los servlets contenidos en *ServletsProyecto* y son estos servlets los encargados de gestionarlas y contestarlas.
  - **SEConnection:** Clase que gestiona la conexión con el Elemento Seguro (SE).
  - **PasswordToServlet:** Clase que muestra una pantalla para confirmar que la contraseña obtenida del Elemento Seguro se quiere mandar al servidor.
  - **Recursos:** Clase que muestra en una lista las reservas realizadas para enviarlas mediante NFCIP o Bluetooth.
  - **Reservas:** Clase que obtiene las reservas realizadas del servidor.
  - **Usuario:** Bean que describe las propiedades de los Usuarios.
  - **Options:** Clase que muestra información sobre las posibles opciones que se obtienen después de leer una etiqueta.
  - **Auxiliar:** Clase que sirve de Auxiliar para gestionar la aplicación, contiene getters y setters, gestión de unas clases a otras…
  - **Información:** Clase que muestra información en la pantalla del móvil.
  - **Cafetería:** Clase que muestra información sobre la cafetería y gestiona sus opciones.
  - **Despacho:** Clase que muestra información sobre los despachos y gestiona sus opciones.
  - **Lab:** Clase que muestra información sobre los laboratorios y las aulas y gestiona sus opciones.
  - **Libro:** Clase que muestra información sobre los libros y gestiona sus opciones.
  - **Ordenadores:** Clase que muestra información sobre las Salas de Ordenadores y gestiona sus opciones.
  - **PIC:** Clase que muestra información sobre los PICs y gestiona sus opciones.
  - **OptionsTutorías:** Clase que muestra información las posibles tutorías.
  - **Fechas:** Clase que muestra información sobre las posibles fechas para las tutorías.
  - **OptionsHora:** Clase que muestra información sobre las posibles horas para las tutorías una vez elegida la fecha.

- **Proyecto *LectorNFCIP***

  - **Target**: Midlet principal que gestiona la conexión NFCIP, al iniciarse muestra la pantalla de contraseña (clase PasswordScreen), después establece la conexión NFCIP (clase NFCIP) y tras recibir los datos comprueba si la contraseña es correcta y muestra la información correspondiente en la pantalla del móvil con la clase Informacion
  - **NFCIP:** Clase que gestiona la comunicación NFCIP.
  - **PasswordScreen:** Clase que muestra la contraseña antes de establecer la conexión NFCIP.
  - **Información:** Clase que muestra informacion en la pantalla del móvil

- **Proyecto *LectorBluetooth***

  - **LectorBluetooth:** Midlet que actúa como lector Bluetooth, busca los dispositivos, muestra una lista con los posibles dispositivos a los que conectarse, establece la conexión con el seleccionado y muestra la reserva recibida.

4.2.2 Aplicación web

Como se ha comentado en el apartado anterior, la aplicación web "Control de Acceso" se ha desarrollado a partir del proyecto *ServletsProyecto,* que también contiene algunos servlets que gestionas las conexiones HTTP recibidas de la aplicación móvil *PFC* como se muestra a continuación:

- **Proyecto *ServletsPFC***

  - **BBDD:** Clase que gestiona las conexiones a la Base de Datos
  - **Usuario:** Bean Usuario
  - **Profesor:** Bean Profesor
  - **Gráfica:** Clase que realiza la gráfica de las estadísticas de los recursos visitados

  - **Servlets**

    - **Prueba:** Servlet que comprueba usuario y contraseña de la aplicación web.
    - **Cerrar:** Servlet que desconecta al usuario de la aplicación web
    - **CafeteriaPFC:** Servlet que gestiona las peticiones referentes a la cafetería procedentes de la aplicación móvil.
    - **DespachoPFC:** Servlet que gestiona las peticiones referentes a los despachos procedentes de la aplicación móvil. LabPFC:
    - **LibroPFC:** Servlet que gestiona las peticiones referentes a los libros **procedentes de la aplicación móvil.**

- **LabPFC:** Servlet que gestiona las peticiones referentes a los laboratorios y las aulas procedentes de la aplicación móvil.
- **PICPFC:** Servlet que gestiona las peticiones referentes a los PICs **procedentes de la aplicación móvil.**
- **OrdenaPFC:** Servlet que gestiona las peticiones referentes a las salas de ordenadores procedentes de la aplicación móvil.
- **ResPFC:** Servlet que gestiona las peticiones referentes a las reservas realizadas procedentes de la aplicación móvil.
- **CafeteriaWeb:** Servlet que gestiona las peticiones referentes a la cafetería procedentes de la aplicación web.
- **LaboratoriosWeb:** Servlet que gestiona las peticiones referentes a los laboratorios procedentes de la aplicación web.
- **AulasWeb:** Servlet que gestiona las peticiones referentes a las aulas procedentes de la aplicación web.
- **Libro:** Servlet que gestiona las peticiones referentes a los libros procedentes de la aplicación web.
- **PICWeb**: Servlet que gestiona las peticiones referentes a los PICs procedentes de la aplicación web.
- **Ordenadores**: Servlet que gestiona las peticiones referentes a las salas de ordenadores procedentes de la aplicación web.
- **Reservas:** Servlet que gestiona las reservas realizadas.
- **Profesores:** Servlet que gestionas los profesores existentes en la base de datos.
- **AnadirTutoría:** Servlet que permite añadir tutorías libres a los profesores desde la aplicación web.
- **EliminarTutoría:** Servlet que permite eliminar tutorías libres a los profesores desde la aplicación web.
- **Estadísticas:** Servlet que gestiona las estadísticas de los recursos visitados.
- Tutorias: Servlet que fija una nueva tutoría.
- **TutoríasProfesorFijadas:** Servlet que gestiona las tutorías fijadas del profesor.
- **TutoriasProfesorLibres:** Servlet que gestiona las tutorías libres del profesor

- **JSP**
  - **index:** JSP principal, que muestra el inicio de la aplicación web.
  - **aulas:** JSP encargado de gestionar las aulas.
  - **cafetería:** JSP encargado de gestionar la cafetería.
  - **laboratorios:** JSP encargado de gestionar los laboratorios.
  - **libros:** JSP encargado de gestionar los libros.
  - **ordenadores:** JSP encargado de gestionar los ordenadores.
  - **pic:** JSP encargado de gestionar los PICs.
  - **profesores:** JSP encargado de gestionar los profesores.
  - **reservas:** JSP encargado de gestionar las reservas.
  - **estadísticas:** JSP encargado de gestionar las estadísticas de los recursos visitados por los alumnos.
  - **tutorías:** JSP que muestra las tutorías disponibles.
  - **tutoríasProfesor:** JSP que muestra las tutorías fijadas de los profesores.
  - **añadirTutorias:** JSP que permite a los profesores añadir tutorías libres.

- **CSS**

  - **sytlePrincipal:** CSS utilizado en el JSP principal, index.jsp.
  - **styleEstadisticas:** CSS utilizado en el JSP que muestra la gráfica con las estadísticas, estadísticas.jsp.
  - **styleWeb:** CSS utilizado en el resto de JSPs.

Capítulo 5

Casos de uso

A lo largo de la memoria se ha mostrado el estudio de las distintas tecnologías que componen el proyecto, los requisitos hardware y software necesarios para su simulación y el código que compone los distintos proyectos que forman la aplicación. Este capítulo se centra en los distintintos casos en los que podemos usar la aplicación.

5.1 Aplicación móvil

La aplicación móvil permite obtener información en distintas ubicaciones de la universidad. A continuación se muestran las distintas ubicaciones y las posibles acciones que podemos realizar en cada una de ellas.

- **Aula 01/ Aula 02**
  - **Consultar horario:** Requiere autenticación. Muestra las horas y las asignaturas que se impartirán.

Figura 5.1 Ejecución de consultar horario en Aula

- **Consultar grupo:** Requiere autenticación. Muestra los grupos que van a tener clase, la hora a la que la van a tener y los profesores que la impartirán.

Figura 5.2 Ejecución de consultar grupo en Aula

- **Consultar profesores:** No requiere autenticación. Muestra los profesores que impartirán clase en esa aula y el departamento al que pertenecen.

Figura 5.3 Ejecución de consultar profesores en Aula

❖ **Despacho 01/Despacho 02**

- **Profesor:** No requiere autenticación. Muestra el profesor al que pertenece el despacho

Figura 5.4 Ejecución de profesores en Despacho

- **Solicitar Tutoría:** Requiere autenticación. En caso de que haya tutorías disponibles muestra una lista con las fechas y al seleccionar una muestra una lista con las horas. Posteriormente muestra si la tutoría se ha podido

reservar con éxito o no (caso en el que alguien haya sido más rápido). También es posible que no haya tutorías disponibles, como se muestra:

Figura 5.5 Ejecución de Solicitar Tutoría en Despacho

❖ **Despacho Inside**

- **<u>Tutorías fijadas:</u>** Requiere autenticación. Este recurso sólo esta disponible para el profesor. Tras autenticarse muestra una lista con las fechas y las horas de las tutorías que han sido solicitadas y el alumno que la solicitó. En caso de no haber tutorías solicitadas lo indica en la pantalla.

Figura 5.6 Ejecución deTutoríasFijadas en DespachoInside

❖ **Laboratorio 01/ Laboratorio 02**

- **<u>Consultar horario:</u>** Requiere autenticación. Muestra información sobre las asignatiuras que se impartirán en ese laboratorio y la hora en las que tendrán lugar.

Figura 5.7 Ejecución deConsultar horario en Laboratorio

- **Consultar grupo:** Requiere autenticación. Muestra los grupos que tendrán clase en el laboratorio, indicando también la hora y qué profesor impartirá la clase.

Figura 5.8 Ejecución deConsultar grupos en Laboratorio

- **Consultar profesores:** No requiere autenticación. Muestra los profesores que impartirán clase en ese laboratorio y el departamento al que pertenece cada uno de ellos.

Figura 5.9 Ejecución deConsultar profesores en Laboratorio

❖ Cafetería

- **Consultar menú:** Se obtiene información sobre el menú del día. Para consultar la información no requiere autenticación, pero para reservar el menú sí. Si no hay menús disponibles no existe la opción de reservar.

Figura 5.10 Ejecución de Consultar menú en Cafetería

- **Plato combinado 1:** No requiere autenticación para consultar de qué esta compuesto pero sí para reservarlo.

Figura 5.11 Ejecución de Consultar plato combinado 1 en Cafetería

- **Plato combinado 2:** Similar a Plato Combinado 1.
- **Plato combinado 3:** Similar a Plato Combinado 1.

❖ **PIC Leganés/Getafe/Colmenarejo**

- **Consultar ubicación:** No requiere autenticación. Muestra la localización del PIC dentro de la universidad y la del Campus.

Figura 5.12 Ejecución de Consultar ubicación en PIC

- **Consultar horario:** No requiere autenticación. Esta opción informa del horario del PIC, tanto los días normales como los horarios especiales de fiestas locales, verano, navidad y Semana Santa.

Figura 5.13 Ejecución de Consultar horario en PIC

- **Contactar:** Como en los casos anteriores, no requiere autenticación por tratarse de un sitio público. Muestra el teléfono y la dirección de contacto.

Figura 5.14 Ejecución de Contactar en PIC

❖ **Sala de Ordenadores 01/Sala de Ordenadores 02**

- **Consultar horario:** Requiere autenticación. Muestra las asignaturas que se impartirán y su horario.

Figura 5.15 Ejecución de Consultar horario en Sala de Ordenadores

- **Consultar aforo:** No requiere autenticación. Esta opción muestra un breve resumen de las características que poseen los ordenadores de la sala y la distribución de los ordenadores disponibles y ocupados en ese momento.

Figura 5.16 Ejecución de Consultar aforo en Sala de Ordenadores

- **Reservar ordenador:** Esta opción requiere autenticación ya que permite reservar un ordenador la sala.

Figura 5.17 Ejecución de Reservar Ordenador en Sala de Ordenadores

❖ **Libro 01/Libro 02**

- **Información del libro:** No requiere autenticación. Muestra un resumen del libro y proporciona información sobre las copias reservadas y disponibles.

Figura 5.18 Ejecución de Información del libro en Libro

- **Asignaturas recomendadas:** Requiere autenticación. Muestra la lista de las asignaturas para las que el libro está recomendado.

Figura 5.19 Ejecución de Asignaturas recomendadas en Libro

- **Reservar:** Está opción requiere autenticación ya que permite reservar el libro. Si hay copias disponibles se realiza la reserva, sino, se indica que no se ha podido llevar a cabo.

Figura 5.20 Ejecución de Reservar en Libro

❖ **Enviar Reservas**

Al seleccionar esta opción, donde es necesaria la autenticación, se accede a la base de datos para obtener un listado con las reservas realizadas y después se da la opción de enviarlo mediante NFCIP o Bluetooth. Esta opción no se puede simular debido a, pero esta implementada en los móviles reales.

Figura 5.21 Envío de reservas

5.2 Aplicación Web

5.2.1 Alumno

En la Aplicación Web para alumnos se muestran las mismas funcionalidades que con la aplicación móvil, con las ventajas de que no necesitan encontrarse en esa determinada ubicación y que la interfaz es más usable.

❖ Consultar de Profesores

Esta opción muestra un listado con los posibles profesores, indicando su departamento y despacho y dando la opción de reservar una tutoría. Al seleccionar esta opción se muestra una pantalla con las tutorías disponibles, indicando su fecha y hora y la opción de solicitarlas. Tras esto se confirma si la reserva se ha realizado con éxito o no.

Figura 5.22 Aplicación Web: Consutar profesores

❖ Consultar salas informáticas

En esta opción se permite consultar las salas deseadas o todas, y se indicará en cada caso las características de los ordenadores de la sala y los ordenadores ocupados y disponibles y, en caso de que queden disponibles, se dará la opción de reservar.

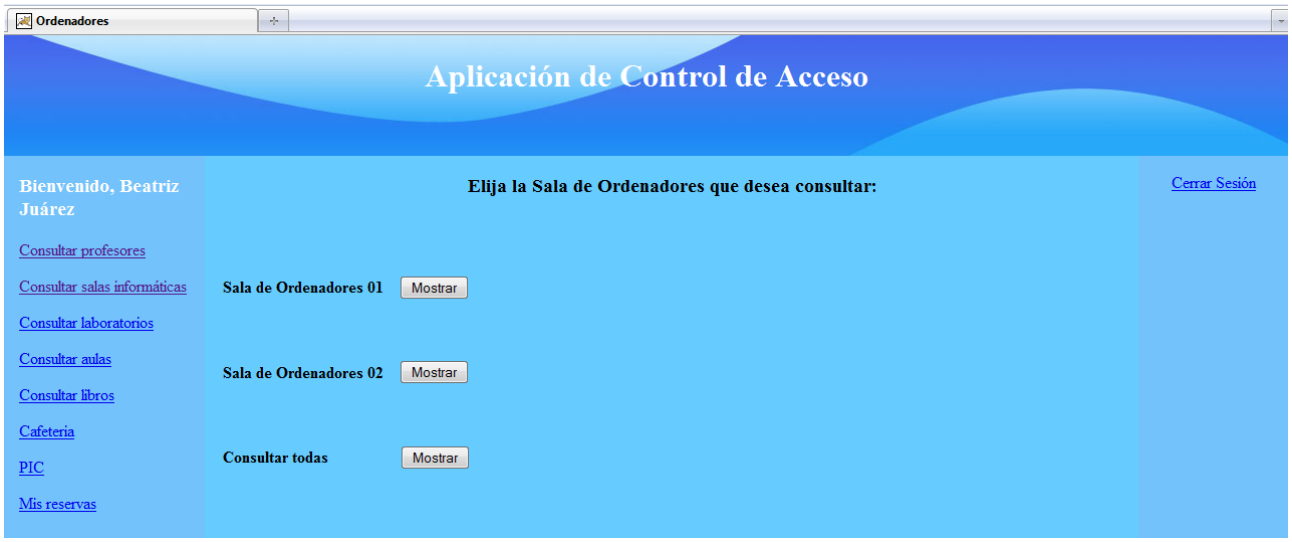


Figura 5.23 Aplicación Web: Consutar salas informáticas

❖ Consultar laboratorios

Se permite consultar un laboratorio específico o todos, y se indicará el horario de cada uno: la hora, la asignatura, el profesor y el grupo. No se permite reservar ordenador ya que en éstos se imparte clase y los de las salas informáticas si se pueden reservar.

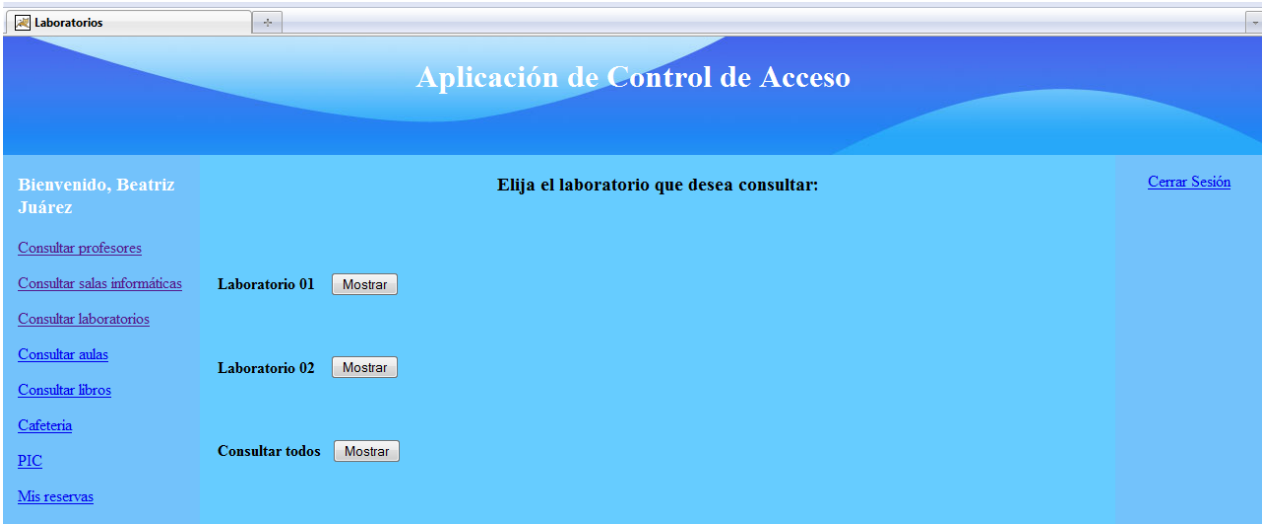


Figura 5.24 Aplicación Web: Consutar salas informáticas

❖ Consultar aulas

Esta opción es similar a la anterior, se permite consultar un aula específica o todas, y se indicará el horario de cada una: la hora, la asignatura, el profesor y el grupo.

Figura 5.25 Aplicación Web: Consutar aulas

❖ Consultar libros

En esta opción se pueden consultar todos los libros: un breve resumen de cada uno, las asignaturas para las que está recomendado y las copias disponibles y reservadas y, en caso de que haya disponibles la opción de reservar.Otra opción es introducir el nombre de la asignatura para la que solicitas el libro y el sistema mostrará el libro adecuado y la opción de reservarlo.

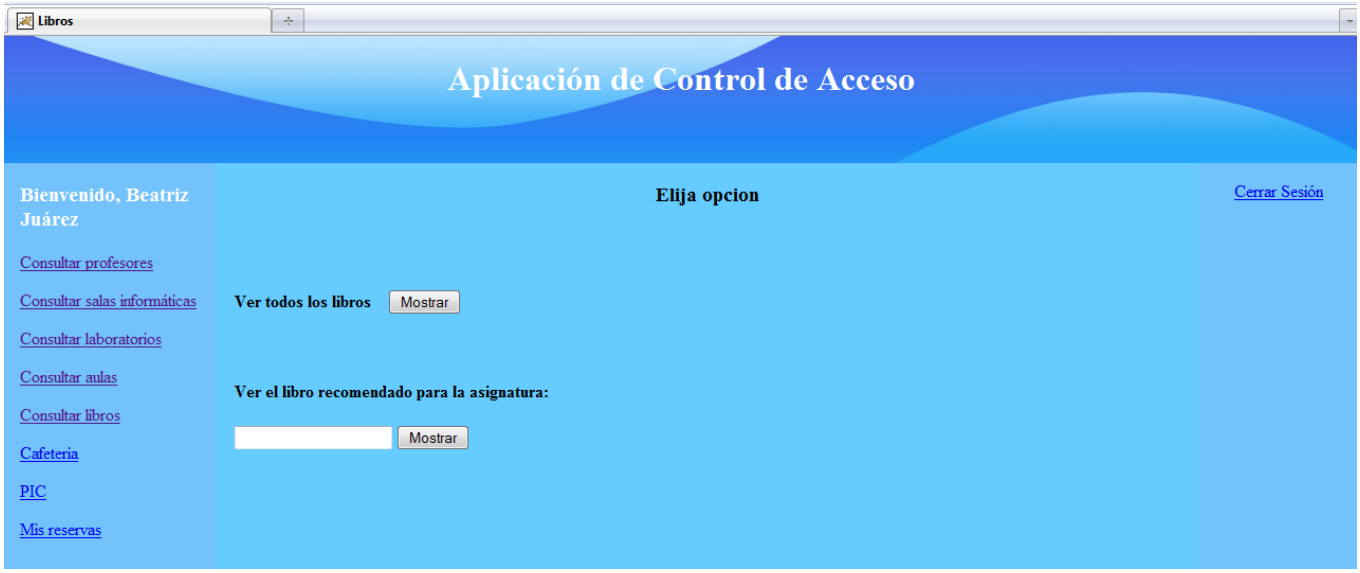


Figura 5.26 Aplicación Web: Consutar libros

❖ Cafetería

En esta opción se da a elegir entre los posibles menús disponibles, se pueden seleccionar y se mostrarán de qué platos están compuestos y cuantos hay libres y cuántos reservardos y, en caso de que queden libres, se podrá reservar.

Figura 5.27 Aplicación Web: Cafetería

❖ **PIC**

Se mostrarán los tres campus de la Universidad y, tras elegir uno, se podrá consultar su ubicación, su contacto o su horario.

Figura 5.28 Aplicación Web: PIC

❖ **Mis reservas**

Se mostrarán las reservas realizadas, con su código de reserva y una breve descripción de la reserva.

Figura 5.29 Aplicación Web: Mis reservas

5.2.2 Profesor

En la Aplicación Web para profeores se añaden más funcionalidades que con la aplicación móvil: se pueden insertar y eliminar tutorías (con la aplicación móvil solo consultarlas) y se pueden ver las estadísticas de los recursos accedidos por los alumnos.

❖ **Consultar tutorías fijadas**

Se mostrarán las tutorías que han sido solicitadas, indicando fecha y hora de cada una así como el alumno que la ha solicitado. En este caso no se da la opción de eliminar la tutoría ya que habría que avisar al alumno, se propone como trabajo futuro.

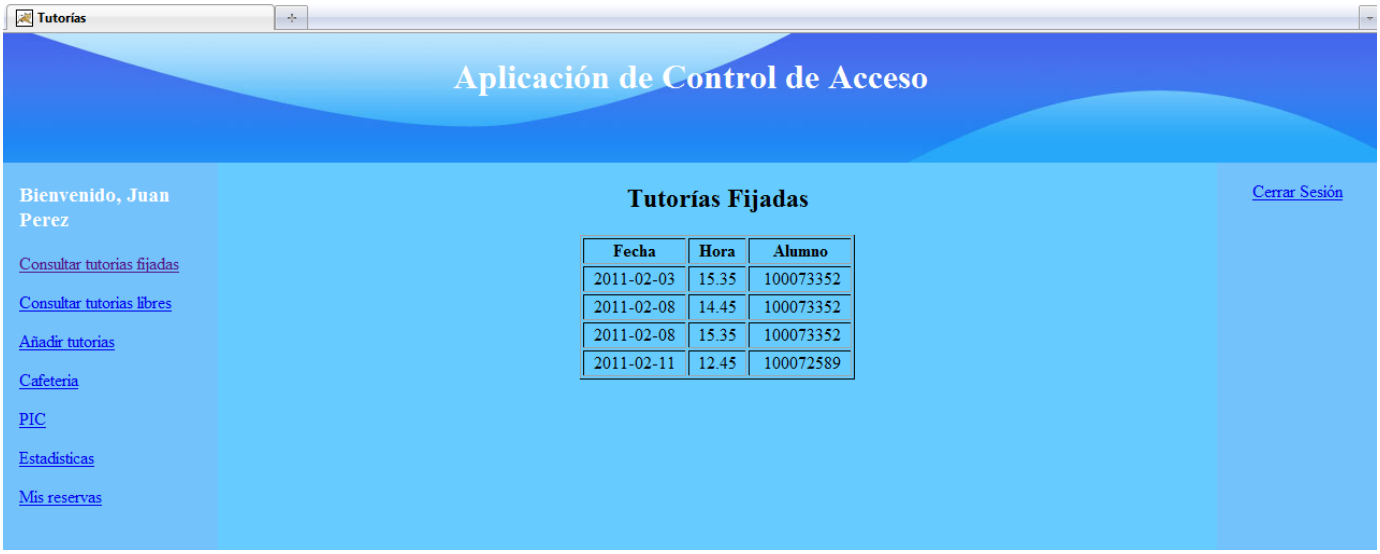


Figura 5.30 Aplicación Web: Consultar tutorías fijadas

❖ **Consultar tutorías libres**

En esta opción, se muestran las tutorías que todavía no han sido solicitadas al profesor y se le da la opción de eliminarlas y que de esta manera, no aparezcan a los alumnos como tutorías disponibles. Esta funcionalidad es añadida, ya que la aplicación móvil no la tiene implementada.

Figura 5.31 Aplicación Web: Consultar tutorías libres

- **Añadir tutorías**

En la aplicación web se añade la funcionalidad de añadir tutorías. El profesor sólo ha de elegir la fecha y la hora y tras pulsar el botón "Añadir Tutoría" aparecerá en *Consultar tutorías libres* y los alumnos podrán solicitarla.

Figura 5.32 Aplicación Web: Añadir tutorías

- **Cafetería**

  Permite ver los tipos de menús disponibles y reservar. Esta opción es similar a las que ya se ha explicado para el alumno (ver sección 2.2).

- **PIC**

  Permite obtener horario, datos de contacto o localización del campus deseado.
  Esta opción es similar a las que ya se ha explicado para el alumnos(ver sección 2.2).

- **Mis reservas**

  Listado con las reservas realizadas: libros, menús… Esta opción es similar a las que ya se ha explicado para el alumno (ver sección 2.2).

- **Estadísticas**

Esta funcionalidad, como la de Añadir y Eliminar Tutorías, sólo está disponible en la aplicación web. Permite visualizar las estadísticas de los recursos accedidos por los alumnos.

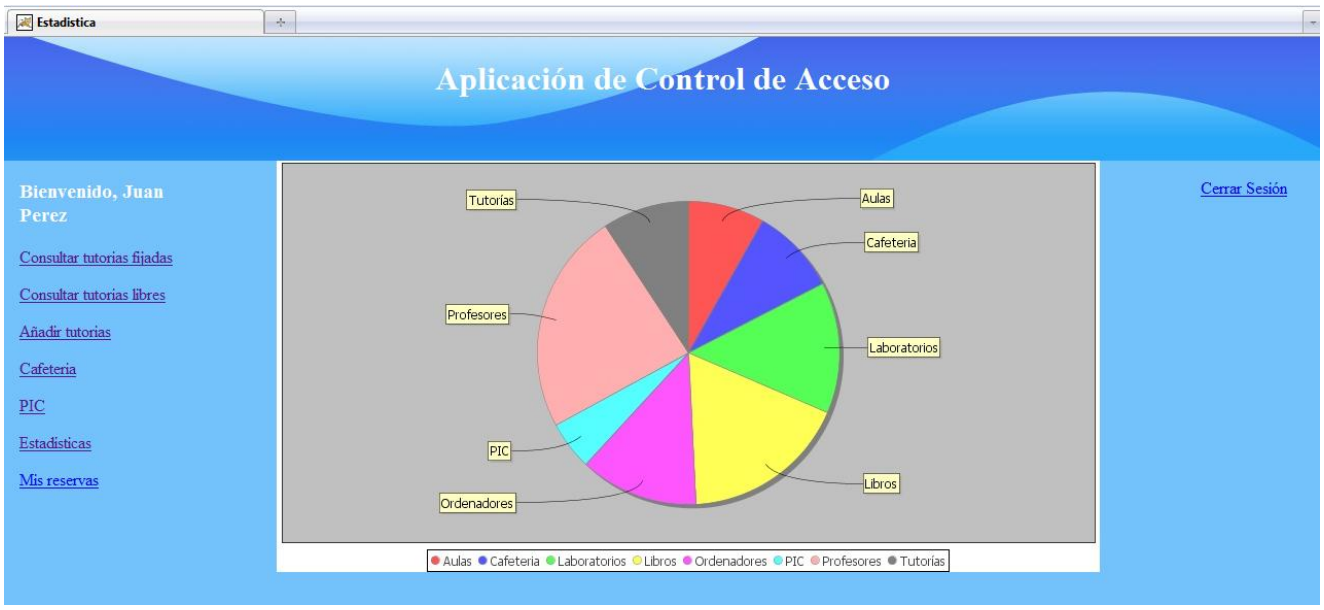


Figura 5.33 Aplición Web: Estadísticas

- **Cerrar Sesión**

Cuando el usuario selecciona esta opción (puede ser alumno o profesor), se le desconecta y se vuelve a mostrar la pantalla de inicio, por lo que si desea seguir interaccionando con la aplicación web ha de volver a autenticarse.

Capítulo 6

Pruebas

El objetivo de este capítulo es mostrar las distintas pruebas realizadas a lo largo de la aplicación para comprobar su funcionalidad así como que cumplía con los objetivos anteriormente mencionados.

6.1 Pruebas en el elemento seguro

En un principio, para poder cargar los Applets en el elemento seguro del Nokia, hubo que desbloquearlo. Para ello, ha de ejecutarse en el teléfono una aplicación con extensión .JAR del Forum Nokia que, concetandose a Internet, desbloquea el elemento seguro.

Tras esto, el primer problema que surgió fue que al intentar cargar el Applet en el elemento seguro obteníamos un "Wrong data" y el Applet no se cargaba. Después de investigar la causa de esto se averiguó que la versión con la que estaba desarrollado el Applet (versión 3.0 de JavaCard) no era compatible, por tanto hubo que volver a crear el Applet esta vez con la versión 2.1.1 y con ello solucionamos el problema.

En la siguiente figura se muestra cómo se soluciona el problema, primero le pedimos al Elemento Seguro a través de la GPShell y del Omnikey cuántos elementos contiene el Applet y nos devuelve cinco elementos. Posteriormente cargamos nuestro Applet y le volvemos a requerir la misma acción; esta vez nos devuelve una lista de seis elementos y entre ellos el nuestro como se muestra a continuación:

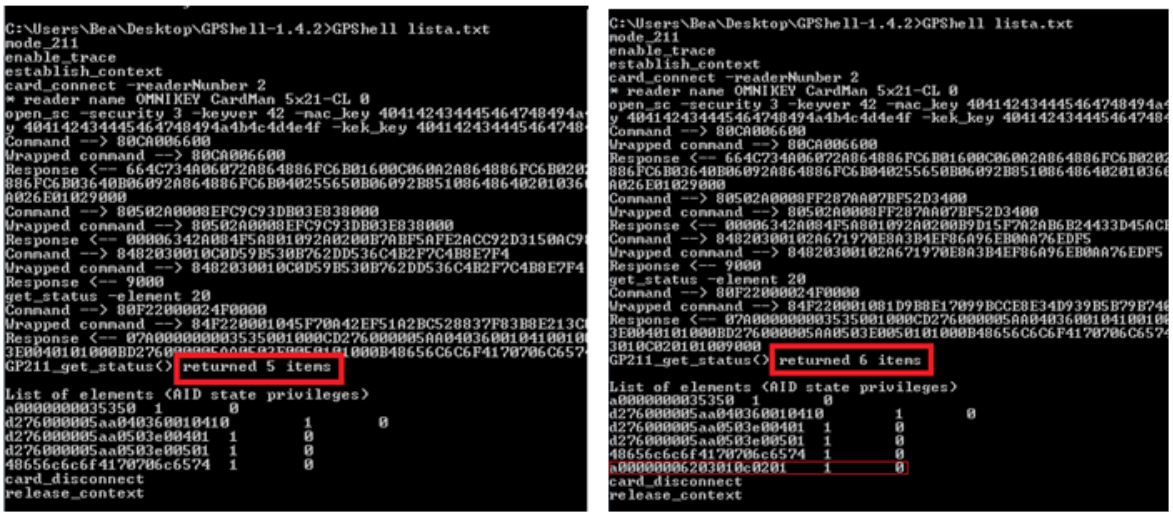


Figura 6.1 Lista de los elementos que contiene el Elemento Seguro

Una vez que el Applet estaba grabado en el elemento seguro nos encontramos con otro problema: se requiere una licencia para desarrollar una aplicación que acceda al elemento seguro. Esta licencia tiene un coste de anual por lo que se decidió buscar una solución alternativa.
En primer lugar, la solución fue crear una clase que simulara el acceso al elemento seguro, pero dado que ya teníamos el Applet creado y grabado en el elemento seguro del móvil real, surgió como solución final desarrollar la aplicación desde el simulador de Nokia pero accediendo al elemento seguro de forma real a través del Omnikey Reader.

6.2 Pruebas de funcionalidad

6.2.1 Funcionalidad de la Aplicación Móvil

- **Escritura de Tags**

Para comenzar ha desarrollar la aplicación móvil, el primer proyecto que se desarrolló fue *WriteTags* pues es el encargado de asignar y grabar en cada Tag su ubicación y las posibles opciones que se pueden comprobar. La funcionalidad de este proyecto está comprobada con la aplicación principal *PFC* pues es la encargada de leer lo escrito en los Tags. El único problema es que al crear un Tag se debe guardar usando *Save Data* pues sino al abrir el simulador la siguiente vez el Tag ya no está.

- **Lectura de Tags**

La aplicación encargada de llevar a cabo esta acción es la desarrollada en el proyecto *PFC*. Esta aplicación lee los Tags y muestra su contenido en la pantalla del móvil por lo que las funcionalidades de escritura/lectura de tags funcionan correctamente.

- **Conexiones HTTP**

Al seleccionar una de las opciones mostradas por el Tag, la aplicación realiza una petición al servlet correspondiente y muestra la respuesta obtenida en pantalla, por lo que la funcionalidad de las conexiones HTTP y del servidor Tomcat quedan demostrados.

El único problema es que en el Nokia 6212 SDK y Windows Vista no funcionan las conexiones HTTP, pero con el simulador Nokia 6131 SDK funciona correctamente.

- **Conexiones Elemento Seguro**

Cuando la aplicación solicita información restringida, el servlet le responde que la autenticación es necesaria y al recibir esta respuesta, la aplicación abre una conexión ISO14443 [41] al elemento seguro para solicitarle la contraseña y este la devuelve correctamente demostrando así el correcto funcionamiento de las conexiones al elemento seguro.

- **Funcionamiento de la Base de Datos**

Las conexiones a la Base de Datos se realizan desde los servlets, bien para autenticar al usuario o para obtener distintos tipos de información. Cuando un usuario envía al servlet la contraseña cifrada en MD5 obtenida de su elemento seguro, el servlet se encarga de mandársela a la Base de Datos para comprobar si es válida o no en función de la respuesta de la Base de Datos. La conexión entre la Base de Datos y los servlets funciona correctamente, verificando así la funcionalidad de ambos.

- **Envío de Reservas mediante NFCIP y Bluetooth**

En este caso nos encontramos un problema: las comunicaciones NFCIP y Bluetooth no son soportadas por el emulador de Nokia, ni el 6131 SDK ni 6212 SDK.

Por tanto, esta funcionalidad se puede comprobar realizando las aplicaciones correspondientes y, sin poder haberlas emulado previamente, cargarlas en el móvil real y comprobar que, tras varios cambios y varias pruebas, funcionan.

- **Reserva de recursos**

Se ha comprobado que cuando un alumno solicita una tutoría y se le confirma que la operación ha sido exitosa, al siguiente alumno que solicita una tutoría con el mismo profesor, si la escoge el mismo día ya no tendrá esa hora disponible. Lo mismo ocurre con los libros, los menús, los ordenadores… cuando un alumno reserva un recurso, el siguiente alumno que quiera reservar observa que hay un recurso menos disponible, y si el del alumno anterior era el último, este alumno ya no puede reservar ya que están todos los recursos agotados. A continuación se muestra un ejemplo:

Figura 6.2 Reserva de recursos disponible y no disponible

6.2.2 Funcionalidad de la Aplicación Web

Ya hemos comprobado con en el apartado anterior el correcto funcionamiento del servidor Tomcat, de las peticiones HTTP y de la Base de Datos, por lo que ahora nos vamos a centrar en otros aspectos de la aplicación web.

- **Autenticación**

Cuando un usuario desea autenticarse, antes de establecer la conexión con la Base de Datos ha de comprobarse que no ha dejado ningún campo vacío (usuario o password). Si este es el caso o se ha rellenado todo completamente, se muestra el error:

Figura 6.3. Pantalla de error que se muestra al fallar la autenticación

- **Reserva de Recursos**

En los casos en los que antes de reservar el recursos se muestra se muestra la cantidad de recursos disponibles, si esta es directamente cero, es decir, que no quedan, no se da la opción de reservar. Si sólo quedara un recurso y lo solicitan dos alumnos, se le adjudicaría al mas rápido de ellos en solicitarlo y al otro se le muestra un mensaje indicando que no ha sido posible continuar con la reserva como en la Figura 5.4

Para verificar que la reserva se ha cursado con éxito, sólo es necesario acudir a la parte inferior izquierda y seleccionar “Mis reservas” dónde se nos mostrará un listado con las reservas realizadas y cada una con un código identificativo único que sirve para identificarla como se muestra en la figura 5.5

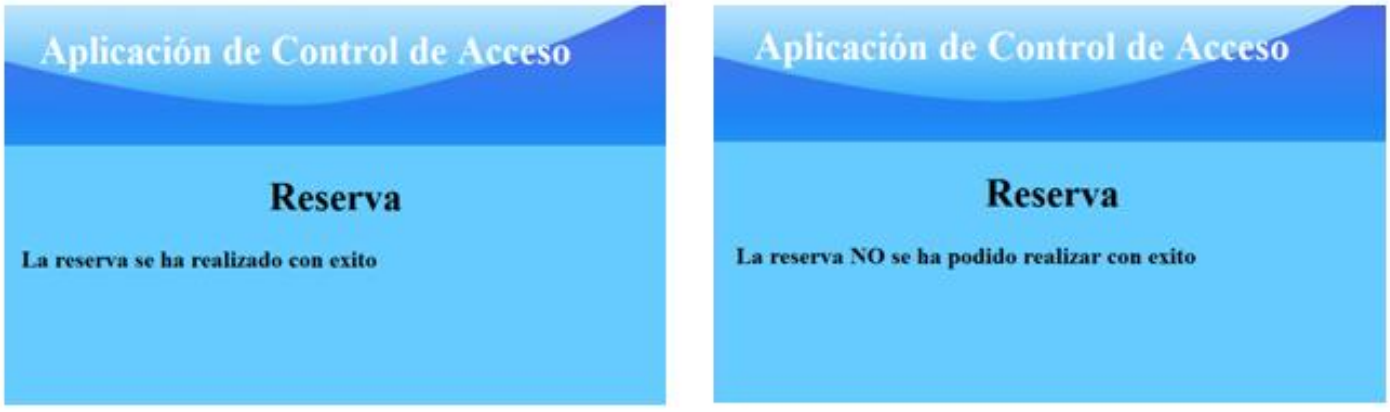


Figura 6.4 Pantallas indicando si la reserva se ha podido realizar o no

Figura 6.5 Pantalla en la que se muestran las reservas realizadas

6.2.2.1 Funcionalidad del profesor

- **Añadir Tutorías**

Los profesores pueden añadir tutorías siempre y cuando lo hagan con el formato concreto que se indica en la aplicación web. Si el formato introducido no es correcto o falta algún dato se mostrará un mensaje de error. Para comprobar que las tutorías se han añadido correctamente, solo es necesario pulsar en la lista que se muestra a la izquierda "Consultar tutorías libres"

Figura 6.6 Adicción de una tutoría y su posterior comprobación

- **Eliminar Tutorías**

Los profesores pueden eliminar tutorías libres y para confirmar que se han eliminado basta con recargar la aplicación y ya no aparece la tutoría eliminada. La funcionalidad de eliminar las tutorías sólo esta disponible en el caso de que no hayan sido solicitadas, ya que de otro modo habría que avisar al alumno, caso que se propondrá en el Capítulo 6 en la sección de proyectos futuros.

Figura 6.7 Eliminación de una tutoría y su posterior comprobación

- **Visualizar gráfica**

Los profesores podrán visualizar una gráfica que mostrará las estadísticas de los recursos visitados por los alumnos: cafetería, laboratorios, aulas…

La gráfica se crea cuando el profesor selecciona la opción "Estadísticas", pero esta gráfica se va modificando según los alumnos vayan solicitando recursos por lo que el profesor deberá actualizar el navergador para mantenerse al tanto de los cambios.

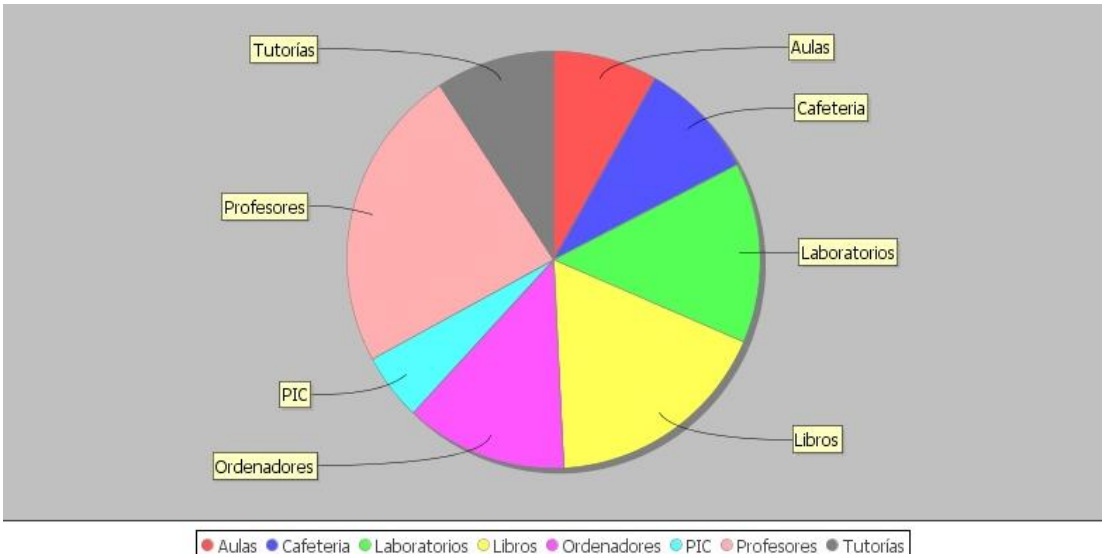


Figura 6.8 Ejemplo de la gráfica de Estadísticas

6.2.2.2 Funcionalidad del alumno

- **Reserva de tutorías**

Cuando un alumno solicita una tutoría, se le muestra un mensaje indicando si ha podido reservarse esa tutoría o por el contrario se ha producido algún error (un alumno la ha solicitado segundos antes o el profesor la acaba de eliminar de sus tutorías disponibles). Una vez que el alumno reciba el mensaje de que la tutoría ha sido fijada con éxito, se puede volver a solicitar las tutorñias disponibles de ese profesor ese mismo día y comprobar que, efectiavamente, la tutoría con la hora que se había solicitado ya no aparece

- **Aumento de visitas de recursos visitados**

Cada vez que un alumno accede a uno de los recursos disponibles: Laboratorios, Aulas, Cafetería… en la Base de Datos se aumentan las visitas a ese recurso. Para comprobar el correcto funcionamiento de esta funcionalidad, basta con observar los datos de la base de datos para comprobar que han aumentado las visitas de cierto recurso o con la gráfica de la Figura 5.8 también se puede comprobar.

Capítulo 7

Conclusiones y trabajos futuros

7.1 Conclusiones

En este proyecto se ha desarrollado una aplicación móvil destinada a dispositivos móviles que permite obtener de una manera eficiente, cómoda y segura información en distintos lugares de la universidad y la realización de reservas.

Para el desarrollo de la aplicación móvil se ha empleado sobretodo la tecnología NFC: para escribir tarjetas, leerlas, enviar reservas mediante NFCIP… El envío de reservas se ha combindo con la tecnología Bluetooth.

Estas tecnologías se eligieron debido al amplio abanico de aplicaciones que nos permiten desarrollar en los dispositivos móviles. Además la mayoría de ellos vienen actualmente equipados con Bluetooth, este no es el caso de NFC pero al ser una tecnología con tantas posibilidades, todo apunta a que, en pocos años, prácticamente todos los teléfonos lo tendrán incorporado.

Durante el estudio de las tecnologías y sobretodo durante el desarrollo de la aplicación observamos las distintas ventajas e incovenientes que nos proporciona cada una de las tecnologías. Tras analizar todos los pros y los contras que han ido surgiendo a lo largo del proyecto, el balance que queda hacer de estas tecnologías es muy positivo.

Desde el punto de vista del desarrollar, el principal inconveniente que cabe destacar es que las comunicaciones Bluetooth y NFCIP no se pueden simular, por lo que

hace el trabajo más lento y tedioso, pero finalmente, cuando las comunicaciones funcionan es cuando se observa que ha merecido la pena. Algunos de los beneficios que cabe destacar para los desarrolladores es que aunque son tecnologías recientes, sobretodo NFC, hay bastante información y ejemplos que son muy útiles para comenzar a desarrollar tu propia aplicación y también hay foros donde encontrar gran cantidad de información.

Desde el punto de vista usuario, la única desventaja que puede surgir es el disponer de un móvil que cuente con ambas tecnologías, o al menos con NFC, porque como ya hemos comentado anteriormente, Bluetooth viene ya en la gran mayoría de los teléfonos. En cuanto a las ventajas desde el punto de vista del usuario son numerosas: las tecnologías son rápidas, eficientes, seguras… y se encuentran en los teléfonos móviles, aparatos que llevamos siempre encima, por lo que son las tecnologías idóneas.

Al haber empleado y desarrollado el proyecto con las dos tecnologías, no sólo observamos las ventajas e inconvenientes que nos ofrecen cada una de ellas, sino que inevitablemente, surge la comparativa entre ellas:

- NFC está estandarizado con el estándar ISO/IEC, mientras que Bluetooth aplica las políticas determinadas en Bluetooth SIG (ver sección 2.3.6)
- La comunicación NFCIP es más rápida que la comunicación Bluetooth. LA conexión entre dos dispositivos NFC se establece rápidamente, mientras que con Bluetooth primero hace una búsqueda de todos los dispositivos con los que puede conectarse y, tras seleccionar el deseado, establece la conexión.
- Una vez establecida la conexión, la velocidad de transmisión de NFC es bastante más baja (424kbits/s como máximo) que la de Bluetooth (2.1Mbits/s como máximo en la versión 2.1)
- En cuanto al alcance de las comunicaciones NFC tiene menor radio de actuacuión. La distancia máxima alcanza los 20centímetros, lo que por un lado se ve como desventaja pues es una distancia muy corta, pero, por otro lado reduce ampliamente la probabilidad de que las comunicaciones sean interceptadas. La máxima distancia Bluetooth alcanza los 30centímetros.
- NFC es compatible con las estructuras exisitentes de la tecnología RFID, al contrario que Bluetooth.
- NFC y Bluetooth v.4 “Low Energy” tienen relativamente el mismo gasto de batería.Sin embargo, en NFC, cuando se trata de móviles o dispositivos que pueden encenderse y apagarse, el consumo de batería aumenta considerablemente.

Sin embargo, es más útil contemplar NFC y Bluetooth no como tecnologías disitntas con sus ventajas e incovenientes, sino intentando aprovechar las ventajas que ofrecen ambas como es el caso de este proyecto. Esto también ocurre en el estándar Bluetooth 2.1, que incorpora “NFC Cooperation”, que permitirá la creación automática de conexiones Bluetooth seguras cuando una interfaz NFC se encuentre disponible.Por ejemplo:

- Unos auriculares con Bluetooth 2.1 pueden conectarse a un móvil con tecnología NFC simplemente acercando los dispositivos.
- Se pueden enviar fotos de un móvil o una cámara de fotos a un marco digital simplemente acercando el teléfono o la cámara al marco.

La otra tecnología utilizada, JavaCard, permite a los desarrolladores construir, probar e implementar aplicaciones y servicios de manera rápida y segura..

Casi cualquier tipo de tarjeta inteligente se puede beneficiar de la tecnología Java Card:

- Tarjets Subscriber Identity Module (SIM), utilizadas en teléfonos móviles en la mayoría de redes inalámbricas.
- Tarjetas de apoyo financiero para operaciones en línea y fuera de línea
- Tarjetas de identidad
- Tarjetas que permiten acceso lógico y físico a diversos lugares, como es el caso de este proyecto
- Bonos de transportes con tarjetas inteligentes

En la mayoría de las redes de telefonía celular, un suscriptor utiliza una tarjeta inteligente comúnmente se llama una tarjeta SIM para activar el teléfono. La tarjeta de la autenticación del usuario y proporciona las claves de cifrado para la transmisión digital de voz. Cuando está equipado con tecnología Java Card, tarjetas SIM también puede proporcionar los servicios transaccionales, como la banca a distancia y la venta de entradas. Cientos de millones de tarjetas SIM basadas en tecnología Java Card estar alimentando servicios innovadores en teléfonos celulares

Los desarrolladores que crean aplicaciones JavaCard disfrutan de todas las ventajas de trabajar en el lenguaje de programación Java:

- Los rendimientos de la programación orientada a objetos ofrecen mayor modularidad y reutilización de código, dando lugar a mayor productividad del programador.
- Las características de protección del lenguaje de programación Java se aplican a los applets de Java Card, la aplicación inflexible de tipos y atributos de protección.
- Potente fuera de la plataforma, herramientas de desarrollo están disponibles.

Entre las posibles ventajas que JavaCard ofrece, podemos destacar:

- **Interoperabilidad:** Los Applets desarrollados con tecnología Java Card se pueden ejecutar en cualquier tarjeta inteligente habilitada con la tecnología Java Card, independientemente del fabricante de la tarjeta y el hardware subyacente.
- **Seguridad:** La tecnología Java Card se basa en la seguridad inherente del lenguaje de programación Java para proporcionar un entorno de ejecución seguro. Creado a través de un proceso abierto, las implementaciones probadas de la plataforma de la industria y las evaluaciones de seguridad garantizarán que los emisores de tarjetas se benefician de la tecnología más capaz y segura disponible en la actualidad.
- **Multi-Application-Capacidad**: La tecnología Java Card permite coexistir a varias aplicaciones de forma segura en una única tarjeta inteligente.
- **Dinámica:** Las nuevas aplicaciones se pueden instalar de forma segura después de que la tarjeta haya sido emitida, por lo que los emisores de tarjetas deben responder de forma dinámica a las necesidades cambiantes de sus clientes.
- **Compatible con las normas existentes:** La API de Java Card es compatible con las normas internacionales para tarjetas inteligentes, tales como ISO7816 o EMV. Las principales normas específicas de la industria como la Plataforma Global y ETSI se refieren a ella.

7.2 Trabajos futuros

En el sector bancario, las tarjetas inteligentes ofrecen a los usuarios un acceso seguro a una amplia gama de red de servicios financieros, incluidos los cajeros automáticos, pago de cuentas, y los peajes del puente. La tecnología Java Card permite a una sola tarjeta inteligente para alojar múltiples aplicaciones financieras, así como ofrecer servicios de terceros, tales como programas de millaje o el comercio en línea seguro.

Otras aplicaciones están disponibles en una amplia variedad, donde la seguridad y la identidad autenticada son importantes, como en el control de acceso a los registros médico o el control de acceso para asegurar las instalaciones como es el caso de este proyecto.

La tecnología Java Card mejorará el acceso del consumidor a los nuevos servicios, el comercio electrónico a través de una serie de aparatos conectados. Los teléfonos celulares y equipos de televisión de pago son ejemplos de mercados en los que la mayoría de los productos ahora ya disponibles incluyen lectores de tarjetas inteligentes, como en este proyecto que se ha utilizado el lector Omnikey.

La tecnología NFC también augura un amplio futuro, para el 2011 aproximadamente 500 millones de teléfonos móviles incorporarán funciones NFC que no sólo serán utilizados para pagar en los comercios, sino también serán usados para acceder a la información disponible en los ‘objetos inteligentes’, en este proyecto los objetos inteligentes son las etiquetas que se situan en los distintos lugares de la universidad y que contiene la información de esa ubicación. En la siguiente figura se muestra un gráfico con la proporción de teléfonos vendidos y cuántos de ellos dispondrán de NFC.

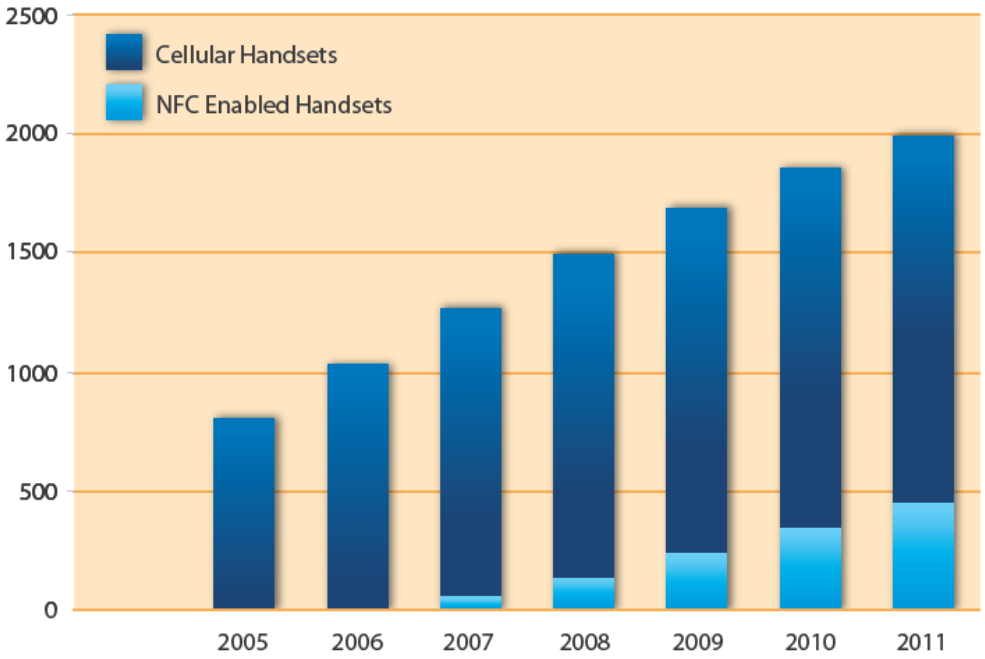


Figura 7.1 Volumen de ventas de móviles con tecnología NFC

La tecnología NFC ya es muy popular en otros países como Japón o Reino Unido, donde el pago con móvil está muy extendido para comprar billetes de metro o realizar pequeñas transacciones. Poco a poco esto se ha ido desarrollando para ampliar el espectro de bienes y servicios que se pueden pagar gracias al chip NFC. Es de esperar que en los próximos años la gama de teléfonos móviles se renueve completamente y que con ello se introduzca la tecnología NFC de forma masiva en el mercado de modo se creen nuevas y sorprendentes aplicaciones que hagan uso de la facilidad y simplicidad del teléfono móvil para mejorar la experiencia del usuario.

Sin embargo, la tecnología NFC presenta varios desafíos que es necesario afrontar para la adopción masiva de esta tecnología. Ejemplo de estos son la necesidad de avanzar en la estandarización de la arquitectura y ubicación del elemento seguro para facilitar el desarrollo de aplicaciones por parte de la comunidad de desarrolladores. El NFC Forum jugará un papel fundamental en la estandarización de la comunicación entre el chip NFC y el procesador base del teléfono, el denominado Host ControllerInterface (HCI). Así mismo, la GSMA deberá definir y estandarizar la comunicación entre la tarjeta SIM y el chip NFC a través del protocolo denominado Single Wire Protocol (SWP) con el fin de conseguir una integración plena del Elemento Seguro dentro de la arquitectura de seguridad. Por último y no menos importante, será necesario crear un modelo de negocio consistente, en que todas las partes involucradas en el ecosistema NFC puedan obtener un retorno de la inversión realizada.

La tecnología Bluetooth se encuentra ampliamente integrada en el mercado: teléfonos móviles, dispositvos manos libres, cámaras de fotos, ordenadores… y no sólo aumenta el número de dispositivos, sino que aumenta el número de aplicaciones desarrolladas con esta tecnología.

7.2.1 Trabajos futuros para este proyecto

Entre los posibles trabajos que se pueden realizar para mejorar este proyecto podemos destacar:

- **Obtención de la licencia necesaria para ejecutar aplicaciones**

  Como hemos comentado anteriormente, la principal limitación de este proyecto consiste en la obtención de una licencia que permita el acceso por parte de las aplicaciones al elemento seguro y así poder ejecutar la aplicación en el móvil. Esta licencia es necesaria pues sin ella esta aplicación sólo puede simularse.

- **Añadir más funcionalidades a la aplicación**

  En la aplicación están registrados las ubicaciones más comunes de la universidad: laboratorios, aulas, despachos, biblioteca, cafetería…pero podrían añadirse más funcionalidades como por ejemplo puede ser la del personal de limpieza: que un profesor pueda escribir con su móvil en un tag a qué hora desea que su despacho sea limpiado, o los servicios de limpieza puedan indicar si una ubicación ha sido ya limpiada o aún no.

- **Desarrollo de Applets con JavaCard v3.0**

En un principio los applets se desarrollaron con la última versión de JavaCard, la v3.0. Sin embargo los ficheros .CAP generados con esta versión no eran compatibles con el elemento seguro del Nokia 6131, como tampoco los soporta ninguna de las versiones existentes de las tarjetas inteligentes.

Por esta razón, en este proyecto no se han podido desarrollar las aplicaciones con las ventajas que esta nueva versión ofrece, entre las que se encuentran la existencia de la clase String y mejoras en el entorno de ejecución que podrían venir bien a las aplicaciones desarrolladas.

Cuando se disponga definitivamente de un entorno de desarrollo y de tarjetas inteligentes adaptadas a Java Card 3.0 sería conveniente realizar como ampliación del proyecto una adaptación de los Applets que se ejecutan en el elemento seguro para que sean desarrollados con esta nueva versión de Java Card.

- **Desarrollo de la aplicación con Nokia 6212 NFC SDK**

Como ya se ha comentado anteriormente, la aplicación se ha tenido que desarrollar con el Nokia 6131 NFC SDK debido a que el SDK del Nkia 6212 no funciona correctamente bajo el sistema operativo Windows Vista.

En teoría, la aplicación desarrollada con el SDK del 6131 debería funcionar correctamente en tanto en el SDK del Nokia 6212 como en el dispositivo real. Sin embargo, estaría bien poder desarrollar la aplicación en el Nokia 6212 SDK ya que posee una mejor interfaz gráfica y además, las comunicaciones NFCIP podrían cifrarse en el Nokia 6212, mientras que el Nokia 6131 no está soportado.

Capítulo 8

Presupuesto

En este apartado se calculará el presupuesto necesario para poder realizar el proyecto. Para calcular los costes del proyecto habrá que tener en cuenta tanto el coste material como el del trabajo de las personas que han participado en su desarrollo.

8.1 Costes de personal

Los costes de personal incluyen los honorarios del Ingeniero Técnico de Telecomunicación en Telemática encargado del desarrollo del proyecto. La duración de este proyecto ha sido de aproximadamente 7 meses. Suponiendo 20 días laborables al mes se obtiene un total de 140 días laborables. Con una jornada laboral de seis horas diarias, la realización del proyecto ha requerido 840 horas aproximadamente. Si se tiene en cuenta la medida de hombres mes (131,25 horas), en la tabla 8.1 se puede ver el resumen de costes directos de personal. Por tanto, el coste total asciende a 50.400€.

Nombre y Apellidos	Categoría	Dedicación (hombre mes)	Coste hombre mes	Coste (Euro)
Beatriz Juárez Gutiérrez	Ingeniero Técnico de Telecomununicación Telemática	6,4	7875	50.400€

Tabla 8.1 Costes de personal

En la siguiente tabla se hace un desglose del coste individual de cada una de las fases de las que consta el proyecto:

Fase	Horas	Coste
Estudio de la Tecnología NFC	102	6120€
Elección de la aplicación	40	2400€
Características de la aplicación	50	3000€
Estudio de alternativas	50	3000€
Implementación de la aplicación	228	13680€
Integración de la aplicación	60	3600€
Creación de la aplicación web	100	6000€
Pruebas	60	3600€
Documentación	150	9000€
TOTAL:	**840**	**50.400€**

Tabla 8.2 Desglose del coste de personal por fases

8.2 Costes de material

Los materiales empleados durante la realización del proyecto han sido los siguientes:

- Un ordenador portatil con sistema operativo Windows Vista, valorado aproximadamente en 1000 euros.
- Dos teléfonos móviles Nokia 6131 NFC, valorado en
- Lector Omnikey Reader 5321, valorado en 100€.
- Conexión a Internet valorada aproximadamente en 40 euros al mes.

Concepto	Unidades	Precio Unitario	% Uso dedicado al proyecto	Dedicación	Periodo de depreciación	Coste imputable
Ordenador portatil	1	1000 €	50%	7	48	72,92€
Nokia 6131 NFC	2	100€	100%	7	36	38,89 €
Lector Omnikey 5321	1	100€	100%	7	36	19,45 €
Internet	1	40€	50%	7	7	140€
Coste Total						**271,26 €**

Tabla 8.3 Costes materiales

Se ha supuesto que la duración de la vida útil del portátil es de 48 meses (4años), por lo que se ha calculado su amortización en 7 meses (el tiempo que se ha necesitado para desarrollar el proyecto). Además, su uso no ha sido únicamente para el proyecto, sólo el 50%, dato que también se ha tenido en cuenta.

Lo mismo ocurre con los teléfonos móviles Nokia y el Lector, que pueden llegar a ser útiles durante 3 años (36meses), pero en este caso su utilización si ha sido única y exclusicavemte para el proyecto.

La conexión a Internet, se ha utilizado durante los 7meses de duración del proyecto, pero no en exclusiva para la realización del mismo, por lo que su uso para el proyecto se estima en el 50%.

8.3 Coste total

El presupuesto total para la realización de este proyecto está constituido por los costes de material y de personal presentados anteriormente. Como se observa en la tabla 8.4, **el presupuesto total asciende a 60.805,51 €€.**

Concepto	Coste
Costes de personal	50.400 €
Costes de material	271,26€
Costes indirectos (20%)	10.134,25€
Presupuesto total:	**60.805,51 €**

Tabla 8.4 Presupuesto total

Anexo I

Manual de Instalación

La finalidad del presente capítulo es que cualquier usuario pueda instalar el proyecto. Para ello, en la primera sección se explicará paso a paso como configurar el entorno de desarrollo y en la segunda sección se explicará como ejecutar los proyectos para poder simular la aplicación.

AI.1 Configuración de la instalación

Para poder ejecutar este proyecto necesitamos **NetBeans 6.9, Tomcat 6.0, Derby y Nokia 6131 NFC SDK.**

- **NETBEANS 6.9**

La versión 6.9 de netbeans se puede descargar en http://netbeans.org/community/releases/69/ y seleccionamos el Netbeans que deseamos descargar que es el que se muestra en la figura A.1. Este Netbeans soporta Java ME necesario para desarrollar las aplicaciones móviles, además, puede integrar la plataforma JavaCard 3 Connected, lo que nos puede ser útil cuando los Applets del elemento seguro se puedan desarrollar con esta versión de JavaCard.

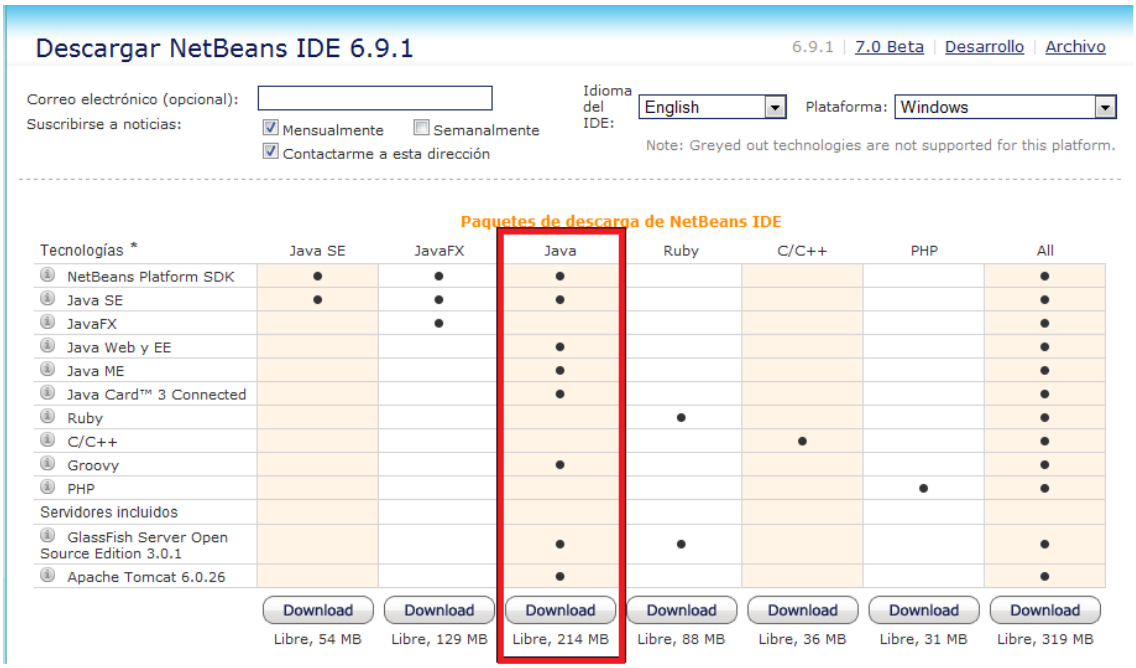


Figura AI.1 NetBeans 6.9 seleccionado

Una vez descargado, descomprimirmos la carpeta y ejecutamos el install.exe que viene y se nos irá guiando por los pasos de instalación. Hay que elegir el directorio donde queremos guardar NetBeans así como el workspace donde se guardarán las carpetas con las aplicaciones creadas.

A lo largo de los pasos de instalación, se pregunta qué completentos se desea instalar, y si seleccionamos Apache Tomcat nos lo ahorraremos después.

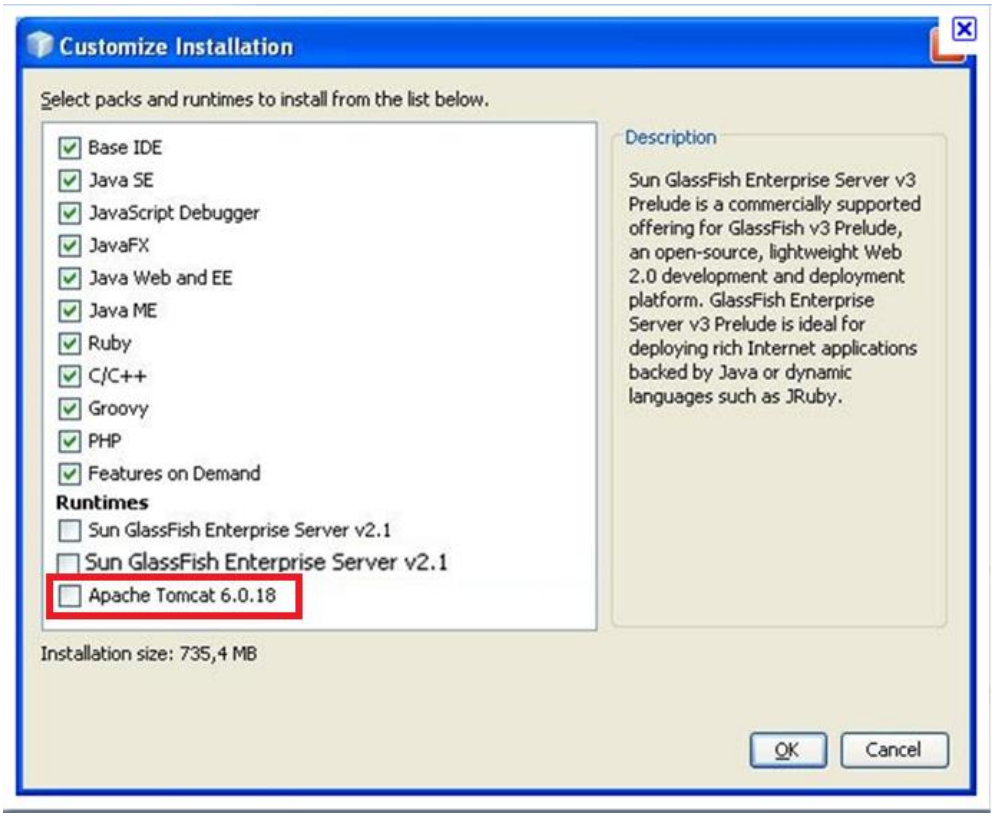


Figura AI.2 Opciones de instalación

Si se ha seleccionado la opción de Apache Tomcat, tendremos que elegir el directorio donde instalarlo, como ocurría con el directorio de NetBeans:

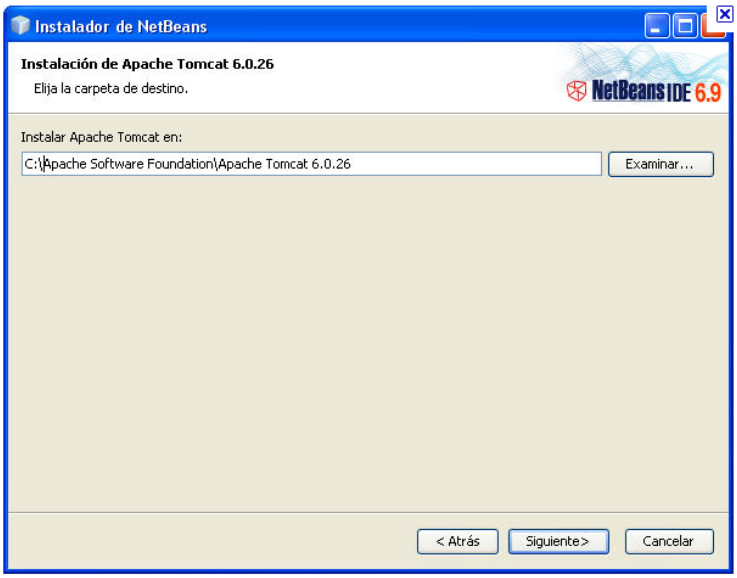


Figura AI.3 Directorio de instalación de Tomcat

Tras finalizar todos los pasos de la instalación, el NetBeans queda instalado en el directorio que le hayamos asiganado y listo para usarse.

❖ **Tomcat 6.0**

El servidor Apache Tomcat debería estar instalado y configurado si se han seguido los pasos establecidos en la instalaciónde NetBeans, saltar a sección siguiente.

En otro caso, podemos descargarlo de http://tomcat.apache.org/download-60.cgi y descomprimir la carpeta.

Una vez que lo tenemos descargado, tenemos que integrarlo en NetBeans. Para ello abrimos el NetBeans y seleccionamos:

Herramientas ⟶ *Servidores* ⟶ *Agregar Servidor* ⟶ *Tomcat 6.0*

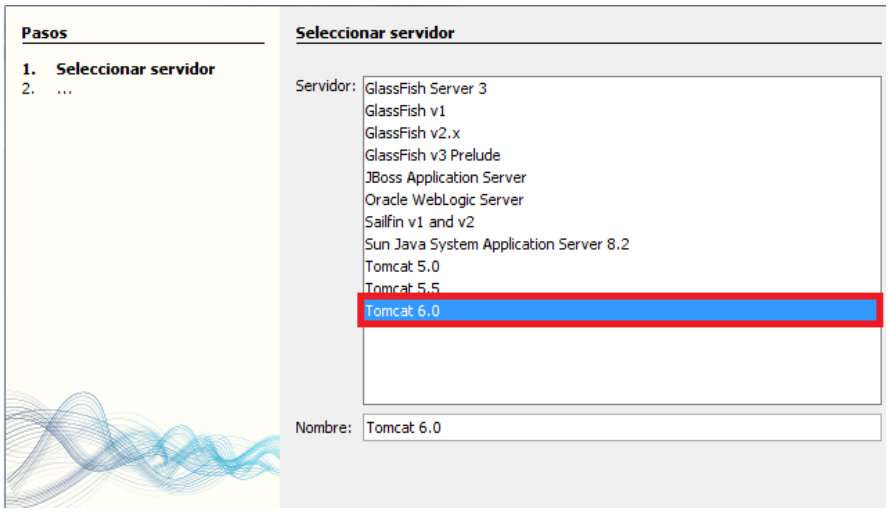


Figura AI.4 Seleccionar Servidor Tomcat 6.0

Una vez elegido Tomcat 6.0 seleccionamos siguiente y, a continuación:

1. Se deberá insertar el directorio dónde se encuentra instalado el servidor.
2. Se deberá establecer un usuario y contraseña

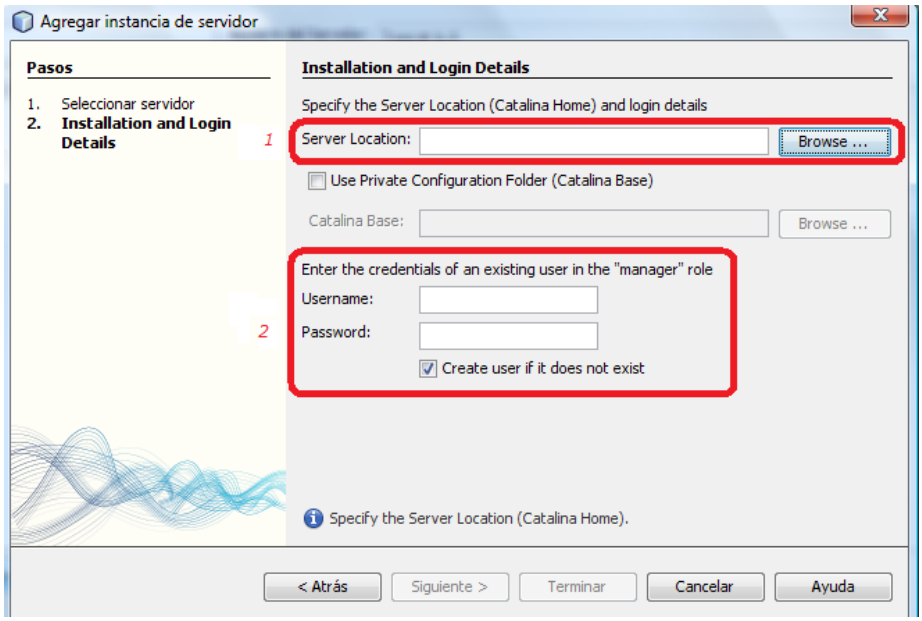


Figura AI.5 Seleccionar directorio y usuario y contraseña

Con esto, el servidor Apache Tomcat queda instalado e integrado en NetBeans.

- **Apache Derby**

Normalmente, Derby debería estar instalado junto con NetBeans, por lo que no sería necesario realizar ninguna configuración adicional.
En caso de no ser así, se puede descargar de http://db.apache.org/derby/releases/release-10.6.2.1.cgi .
Como anteriormente se ha comentado en la instalación de Tomcat, una vez que la carpeta se haya descargado, ha de descomprimirse e integrase en NetBeans.Para ello hay que abrir NetBeans y seleccionar:

Ventana ⟶ *Prestaciones* ⟶ *Base de Datos* ⟶ *Nueva Conexión de Base de Datos*

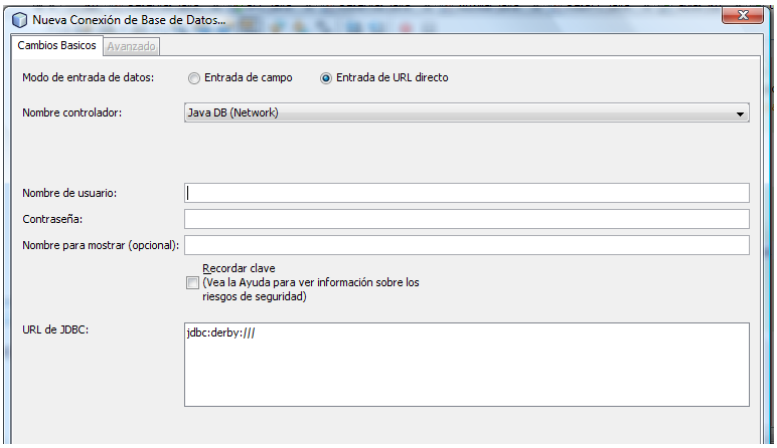


Figura AI.6 Crear Nueva Conexión a la Base de Datos

- **Nokia 6131 NFC SDK**

Puede descargarse de www.forum.nokia.com/info/sw.nokia.com/id/ef4e1bc9-d220-400c-a41d-b3d56349e984/Nokia_6131_NFC_SDK.html . Se descomprime la carpeta y se ejecuta la aplicación *setup.exe.*
A continuación, aparecen distintitos pasos para la instalación, dónde debemos seleccionar queé software instalar, como se muestra en la siguiente figura:

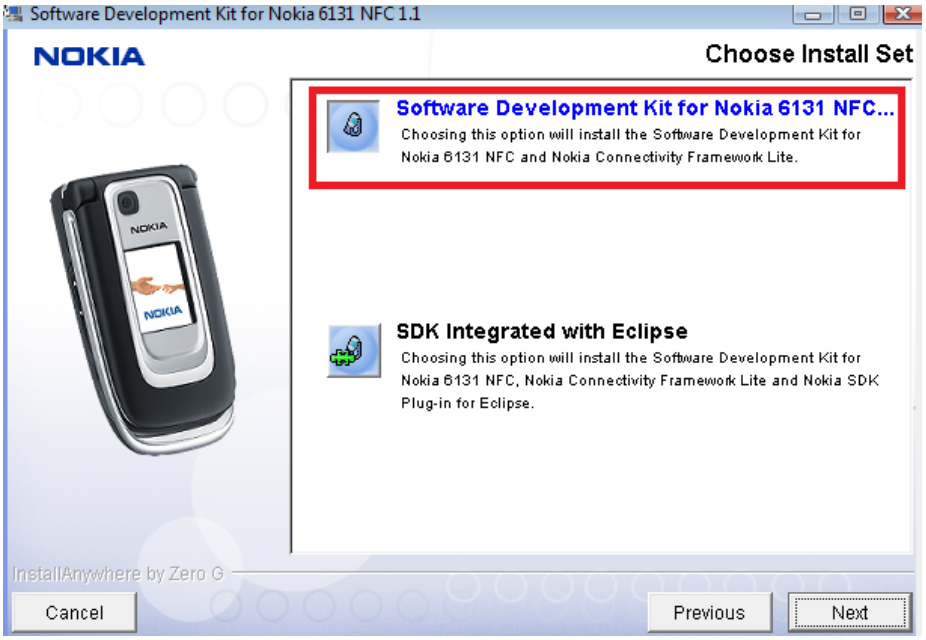


Figura AI.7 Elección del software a instalar

Posteriormente elegimos el directorio dónde instalar el SDK, que por defecto es en C:\Nokia\Devices. Podemos seleccionar ese u otro y pulsar *Next:*

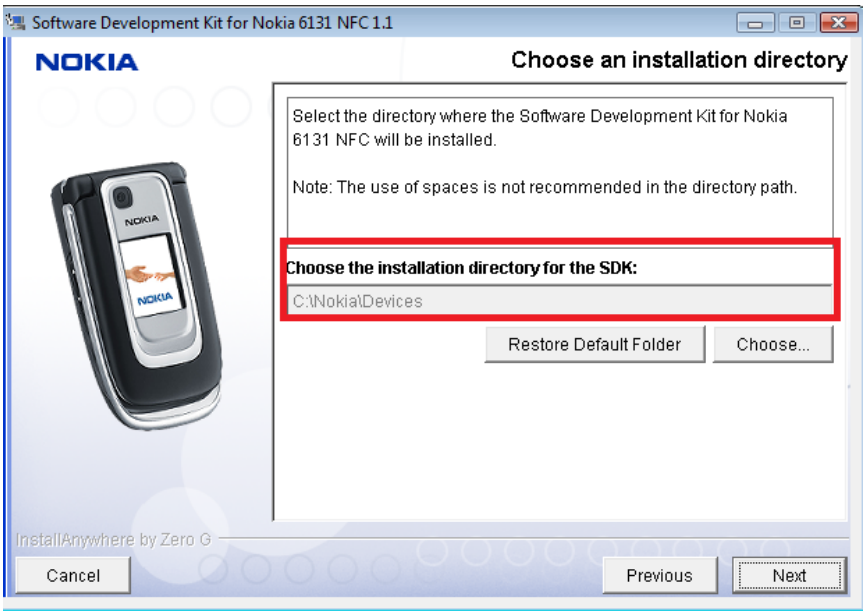


Figura AI. 8 Elección del directorio deseado

Por último, se muestra un resumen del software a instalar y el lugar dónde se va a instalar, ha de comprobarse que todo sea correcto y se selecciona *Install* para completar la instalación como se muestra en la Figura A.9:

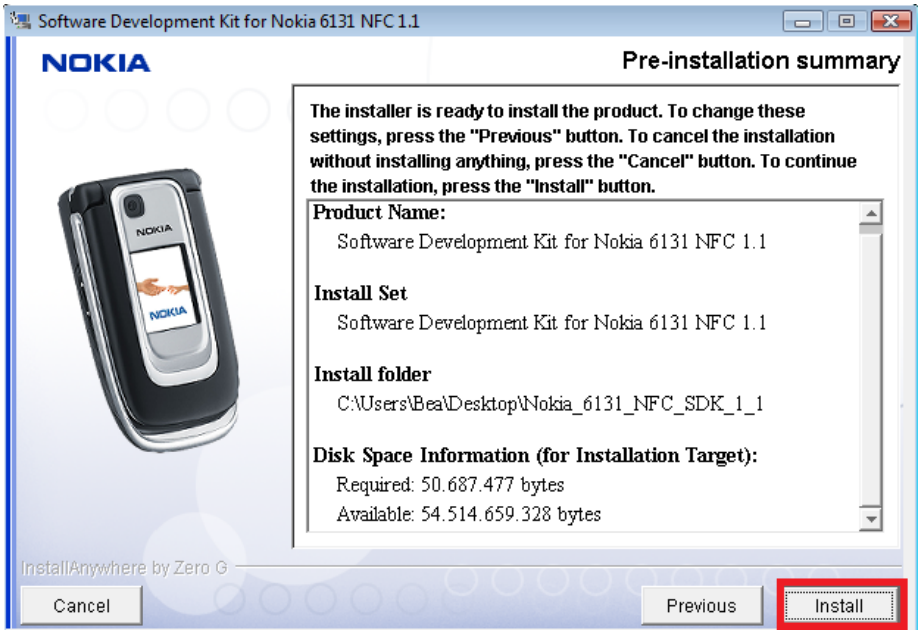


Figura AI.9 Instalar el software deseado en la ubicación seleccionada

Una vez que está instalado queda el último paso: intregarlo con NetBeans. Para ello, en NetBeans seleccionamos:

Herramientas ⟶ *Plataformas Java* ⟶ *Añadir Plataforma.*

Seleccionamos la opción que aparece en la Figura A.10:

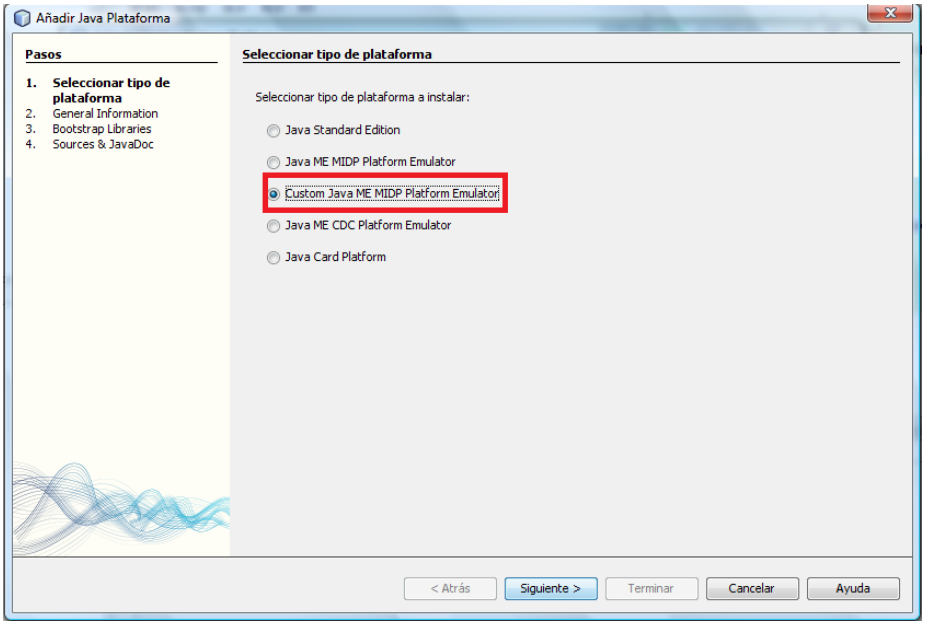


Figura AI.10 Seleccionamos Custom JavaME MIDP Platform Emulator

A continuación se ha de introducir el directorio donde se encuentre instalado el SDK, que se corresponde con el introducido en la Figura A.8 y tras esto, la plataforma ya está integrada.

❖ **Instalar Applet en el Elemento Seguro**

Para poder instalar el Applet en el Elemento Seguro se necesita GPShell 1.4.2 y el Omnikey Reader.
En primer lugar es necesario instalar la aplicación UnlockMIDlet.jar en el telefóno. Esta aplicación se conectará a Internet y, tras seguir los pasos que se indican en la pantalla del móvil, el Elemento Seguro del móvil se ha desbloqueado.

- **GPSHell 1.4.2:** Puede descarse de http://sourceforge.net/projects/globalplatform/files/GPShell/GPShell-1.4.2/GPShell-1.4.2.zip/download. Una vez descargado la carpeta ha de descomprimirse pero no es necesario instalarlo.

Para instalar el Applet en el Elemento Seguro del móvil ha de copiarse el archivo .CAP dentro de la carpeta GPShell y tras esto, se abre una ventana CMD y, situándonos en el directorio donde se encuentren el .CAP y el fichero helloInstall.txt que contendrá los comandos para instalar el fichero .CAP, ejecutamos GPShell como se muestra:

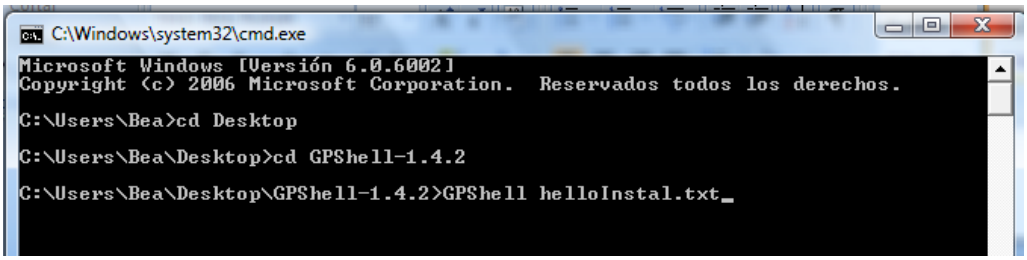


Figura AI.11 Instalación del Applet en el Elemento Seguro con GPShell

Para poder llevar a cabo la instalación es necesario tener el lector Omnikey conectado al ordenador y el Nokia 6131 NFC real situado encima del lector para que se pueda grabar el Applet.

AI.2 Ejecución de la aplicación

Una vez que ya se tengan instalados todos los elementos necesarios, para ejecutar la aplicación, para ejecutarla debemos abrir NetBeans y seleccionar:

Archivo ⟶ *Abrir Proyecto* ⟶ *Seleccionar el proyecto*

Esto debemos hacerlo para los proyectos ServletsProyectos, WriteTags y PFC. Para estos dos últimos debemos seleccionar las propiedades del proyecto y en la pestraña Platform seleccionar la plataforma del Nokia 6131 NFC SDK como se muestra en la Figura A.12.

En la aplicación ServletsProyecto basta con pinchar con el botón derecho del raton y seleccionar la opción *Deploy*.

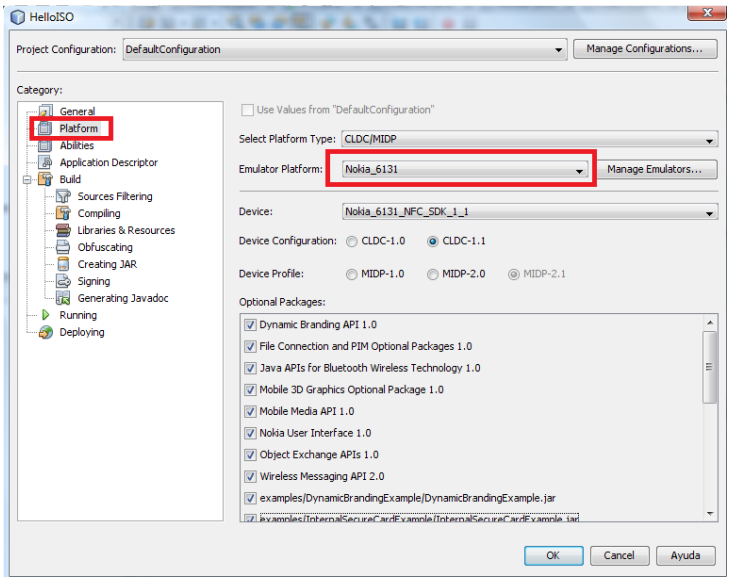


Figura AI.12 Configuración de la Plataforma de los Proyectos

Una vez que ya está todo instalado y configurado, es necesario iniciar el Tomcat (se pincha sobre él y se selecciona Start) y la Base de Datos (se pincha sobre ella y se selecciona conectar).
Para ejecutar la aplicación *PFC* simplemente la seleccionamos y elegimos *RUN*.
Aparecerá el emulador Nokia 6131 NFC y se podrá interaccionar con la aplicación.

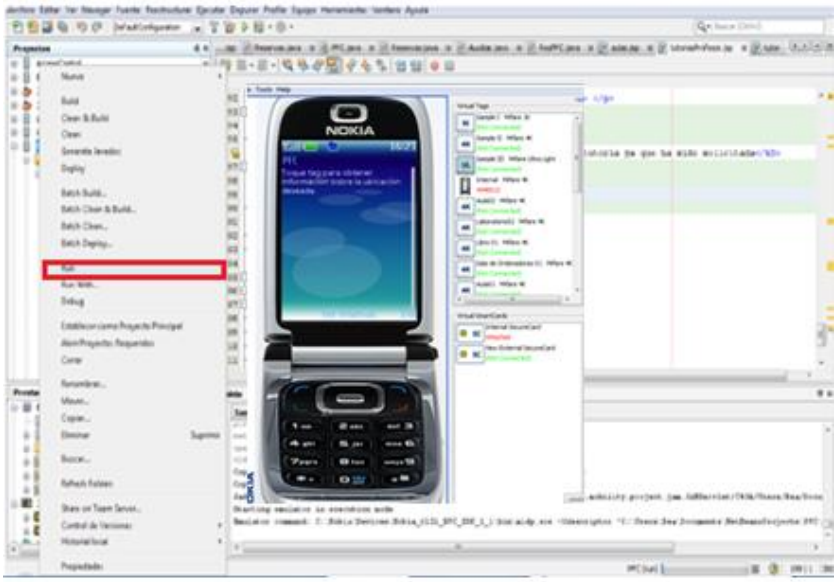


Figura AI.13 Ejecución de la aplicación móvil

Si se desea iniciar la aplicación web seleccionamos *ServletsProyectos* y seleccionamos Ejecutar, tras esto, se abrirá un explorador web dónde se podrá interactuar con la aplicación.

Anexo II

Manual de Usuario

En apartados anteriores se ha especificado cuáles son los requisitos software necesarios para poder desarrollar la aplicación, así como su instalación. Con este manual se trata de explicar cómo simular la aplicación, ya que en un entorno real no se puede ejecutar por la falta de lincencia.

AII.1 Aplicación móvil

En primer lugar, ha de ejecutarse el proyecto PFC tal y cómo se explicó en la sección AI.2. Es preciso recordar que para poder ejecutarla es necesario tener el Omnikey Reader y el Nokia 6131 NFC con el Applet en su Elemento Seguro. Para que durante la simulación se acceda al Elemento Seguro del Nokia real, es necesario configurarlo seleccionando:

1. InternalSecureCard
2. External Omnikey Reader

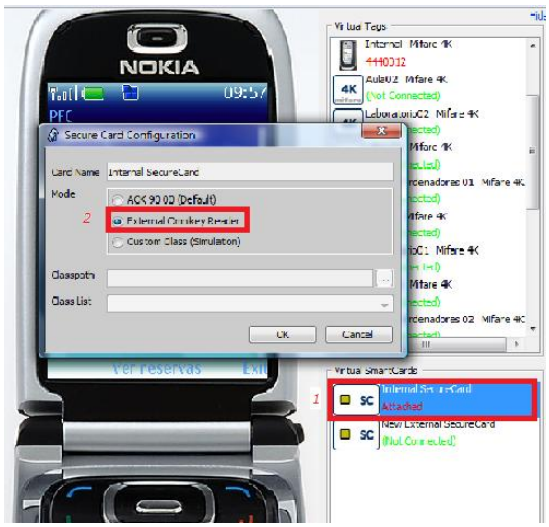


Figura AII.1.Seleción Omnikey

1.1 Simulación Tag

Para simular las distintas ubicaciones posibles, simplemente basta con arrastrar uno de los tags situados a la derecha del emulador, en la pantalla "Virtual Tags", como se muestra en la Figura AII.2

Tras esto nos aparecerán las posibles opciones que se permite realizar en esa determinada ubicación.

A lo largo de este manual se tomará como ejemplo el Tag "Sala de Ordenadores 01" que, como su propio nombre indica, simula la ubicación de una Sala de Ordenadores.

El resultado que obtenemos al arrastrar el Tag se muestra en la Figura AII.3.

Figura AII.2 Simulación del Tag

Figura AII.3 Resultado de la simulación

1.2 Conexión con el servidor y Base de Datos

Tras elegir una de las posibles opciones que aparecen en la pantalla, se abrirá una conexión al servidor para enviar la solicitud deseada y, si la petición deseada no requiere autenticación, el servidor se conectará a la base de datos para obtener la información. A continuación, el servidor envía los datos recibidos al móvil. La secuencia que se sigue es la siguiente:

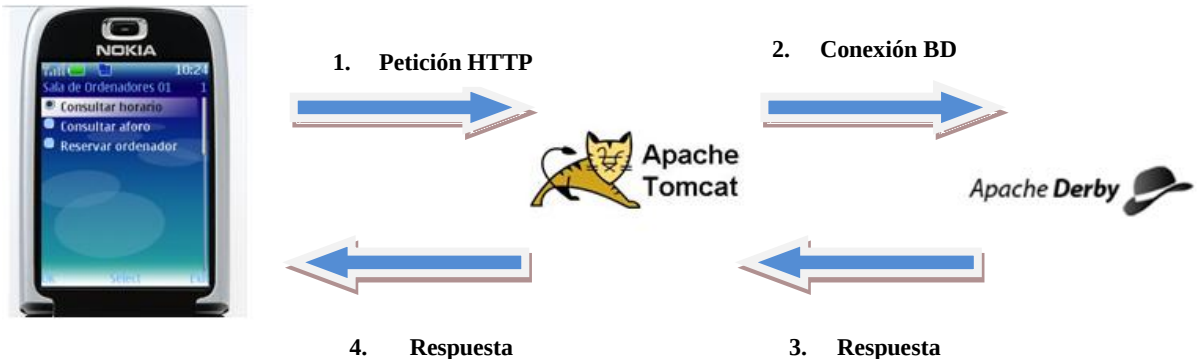


Figura AII.4 Solicitud sin autenticación

En el ejemplo del Tag “Sala de Ordenadores 01”, la opción “Consultar aforo” sigue este esquema: no requiere autenticación y nos muestra una breve información de esa sala en concreto así como la cantidad de ordenadores y cuántos están libres y cuántos ocupados.

Figura AII.5 Ejemplo de simulación

1.3 Acceso al Elemento Seguro

En algunos casos, la opción solicitada requiere autenticación para darle mayor seguridad a la aplicación y que sólo accedan a información restringida los usuarios debidamente autenticados. Cuando el móvil solicita la información al servidor, este comprueba si requiere autenticación y, si es el caso, se lo indica al móvil que tendrá que conectarse al Elemento Seguro para obtener la contraseña cifrada y se la enviar al servidor. Éste se conecta a la base de datos para comprobar si es válida y cuando obtiene el resultado de la base de datos, si es válida envía la petición solicitada a la base de datos y envía la respuesta al móvil, como en el apartado anterior. En caso de no ser válida se lo indica al móvil y deniega el acceso a la información.

- **Autenticación válida**

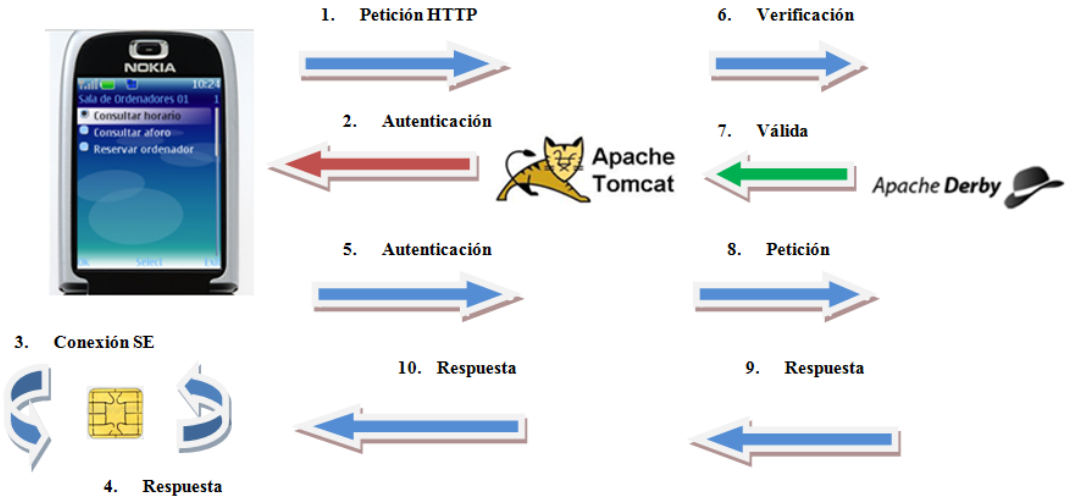


Figura AII.6 Esquema de autenticación válida

En el ejemplo del Tag “Sala de Ordenadores 01”, la opción “Reservar ordenador” sigue este esquema: requiere autenticación y nos permite reservar un ordenador en esa sala.

Figura AII.7 Autenticación correcta

- **Autenticación inválida**

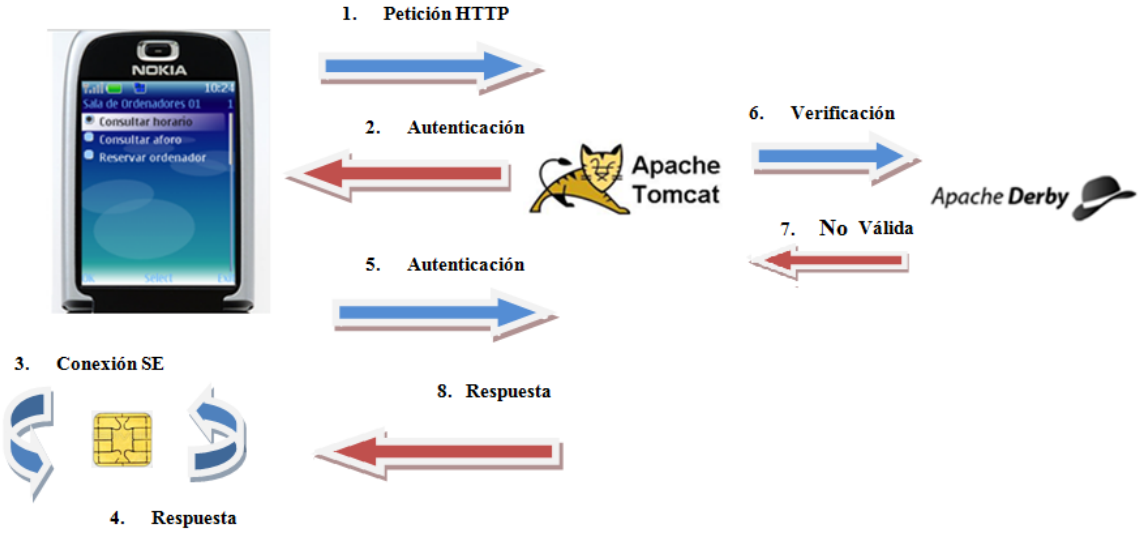


Figura AII.8 Esquema de autenticación inválida

En el ejemplo del Tag “Sala de Ordenadores 01”, la opción “Reservar ordenador” sigue este esquema: requiere autenticación y la que enviamos es incorrecta **No**

Figura AII.9 Autenticación incorrecta

AII.2 Aplicación web

En la aplicación web existen también dos roles: alumno y profesor. Para ejecutarla ha de inciarse Tomcat y Derby (ver sección AI.2) y seleccionar *ServletsProyectos* y *Ejecutar*, como se muestra en la siguiente figura.

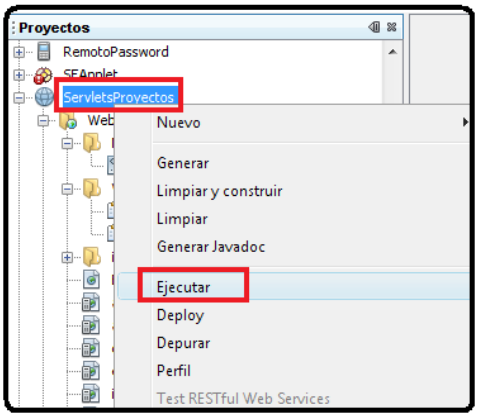


Figura AII.10 Ejecución de la Aplicación web

2.1 Inicio

El inico de la aplicación es común para profesores y alumnos, han de introducir su identificador de usuario y su contraseña en los campos situados en la parte superior izquierda de la aplicación.

Figura AII.11 Pantalla de inicio de la Aplicación Web

Si la autenticación es correcta, se mostrará una pantalla de acuerdo a los recursos a los que se puede acceder según su rol. Si el usuario no se autentica correctamente se le mostrará error.

Figura AII.12 Error durante la autenticación

Una vez que la autenticación ha sido correcta, se mostrará la pantalla de inicio correspondiente, según sea alumno o profesor, posibilitándole así las distintas opciones que están accesibles para cada caso:

- ***Pantalla de inicio para Alumno***

Figura AII.13 Inicio Alumno

- ***Pantalla de inicio para Profesor***

Figura AII.14 Inicio Profesor

Una vez que el usuario este correctamente autenticado, el modo de interacción es similar al de una aplicación web normal, basta con seleccionar la opción deseada y se mostrará la pantalla correspondiente.

2.2 Cierre de Sesión

Cuando el usuario desee abandonar la aplicación, basta con seleccionar la opción que aparece en la parte superior derecha "*Cerrar Sesión*". Si posteriormente se desea volver a consultar alguna de las opciones será necesario volver a autenticarse.

Glosario

AES *Advanced Encryption Standard*
AID *Application IDentifier*
APDU *Application Data Unit*
API *Application Programming Interface*
CDC *Connected Device Configuration*
CLA *CLAss, byte de clase de una APDU*
CLDC *Connected Limited Device Configuration*
CVM *Compact Virtual Machine*
DES *Data Encryption Standard*
GIAC *General Inquiry Access Code*
HTTP *HyperText Markup Language*
IAC *Inquiry Access Code*
INS *INStruction, byte de instrucción de una APDU*
IrDA *Infrared Data Association*
ISO *International Organization for Standarization*
JAR *Java ARchive*
J2EE *Java Plataform Enterpise Edition*
J2ME *Java Plataform Movile Edition*
J2SE *Java Plataform Standar Edition*
JCRE *Java Card Runtime Enviroment*
JCVM *Java Virtual Machine*
JDBC *Java DataBase Connectivity*
JSP *JavaServer Pages*
KVM *Kilobyte Virtual Machine*
LC *Command Length, parámetro de una APDU*

LE	*Length Expected, parámetro de una APDU*
MIDP	*Mobile Information Device Profile*
NDEF	*NFC Data Exchange Format*
NFC	*Near Field Communication*
NFCIP	*NFC Interface and Protocol*
PIN	*Personal Identification Number*
RID	*Resource IDentifier*
RFID	*Radio Frequency IDentification*
SIG	*Bluetooth Special Interest Group*
SIM	*Subscriber Identity Module*
SQL	*Structured Query Language*
SW	*Status Word*
SCWS	*Smart Card Web Server*
PID	*Porpietary Identifier*

Referencias

[1] RFID
http://es.wikipedia.org/wiki/RFID Octubre-2010

[2] De RFID aNFC
http://www.differencebetween.net/technology/difference-between-rfid-and-nfc/
Octubre-2010

[3] Comparación NFC con otras tecnologías
http://www.3gtech.info/tag/nfc-comparison Septiembre-2010

[4] Modos de funcionamiento NFC
http://www.gruposodercan.es/archivos/documentos_contenidos/1868_1.saber_mas_2.PDF Agosto-2010

[5] Arquitectura de dispositivo móvil NFC
http://www.terra.es/personal/ccossio/tecnologiaNFC_5.htm Noviembre-2010

[6] NFCForum
http://www.nfc-forum.org/home/ Julio-2010

[7] NFC Data Exchange Format (NDEF).
http://www.nfc-forum.org/specs/spec_list/#ndefts Junio-2010

[8] NFC Record Type Definition (RTD).
http://www.nfc-forum.org/specs/spec_list/#rtds Diciembre-2010

[9] NFC Smart Poster Record Type Definition.
http://www.nfc-forum.org/specs/spec_list/#rtds Diciembre-2010

[10] NFC Text Record Type Definition.
http://www.nfc-forum.org/specs/spec_list Diciembre-2010

[11] NFC URI Record Type Definition.
http://www.nfc-forum.org/specs/spec_list Diciembre-2010

[12] Modos de funcionamiento NFC
http://es.wikipedia.org/wiki/Near_Field_Communication Septiembre-2010

[13] Etimologia Bluetooth
http://blackinside.wordpress.com/2007/03/28/bluetooth/ Diciembre-2010

[14] Comparación Bluetooth con otras tecnologías
http://www.bluetooth.com/English/Technology/Works/Pages/Compare.aspx Diciembre-2010

[15] Bluetooth SIG
http://www.bluetooth.com/English/SIG/Pages/default.aspx Diciembre-2010

[16] Piconet
http://es.wikipedia.org/wiki/Piconet Enero-2011

[17]Scatternet
http://en.wikipedia.org/wiki/Scatternet Enero-2011

[18]Protocolo Bluetooth L2CAP
http://en.wikipedia.org/wiki/Bluetooth_protocols#Logical_link_control_and_adaptation_protocol_.28L2CAP.29 Diciembre-2010

[19]Protocolo Bluetooth SDP
http://en.wikipedia.org/wiki/Bluetooth_protocols#Service_discovery_protocol_.28SDP.29 Octubre-2010

[20] Protocolo Bluetooth RFCOMM
http://en.wikipedia.org/wiki/Bluetooth_protocols#Radio_frequency_communication_.28RFCOMM.29 Octubre-2010

[21] Protocolo Bluetooth OBEX
http://en.wikipedia.org/wiki/Bluetooth_protocols#Object_exchange_.28OBEX.29
Octubre-2010

[22] Java Platform Standard Edition
http://en.wikipedia.org/wiki/Java_Platform,_Standard_Edition Junio-2010

[23] Java Platform Enterprise Edition
http://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition Junio-2010

[24] Java Platform Micro Edition
http://en.wikipedia.org/wiki/Java_Platform,_Micro_Edition Junio-2010

[25] Configuración CLDC (Connected Limited Device Configuration)
http://java.sun.com/products/cldc/ Diciembre2010

[26] Configuración CDC (Connected Device Configuration)
http://www.oracle.com/technetwork/java/javame/tech/index-jsp-139293.html Diciembre-2010

[27] KVM (Kernel Based Virtual Machine)
http://www.linux-kvm.org/page/Main_Page Diciembre-2010

[28]CVM (Compact Virtual Machine)
http://www.developer.com/java/j2me/article.php/1378971 Diciembre-2010

[29] Perfiles
http://www.slideshare.net/ingeniods/java-a-tope-j2-me-java-2-micro-edition Diciembre-2010

[30] MIDP
http://www.oracle.com/technetwork/java/index-jsp-138820.html Junio-2010

[31]Espcificación JavaCard 2.2.1
http://java.sun.com/javacard/specs.html. Julio-2010

[32] JavaCard Technology APIs
http://java.sun.com/javacard/reference/docs/index.jsp Julio-2010

[33] JavaCard Applet
http://es.wikipedia.org/wiki/Java_Card Julio-2010

[34]APDU
http://www.jguru.com/faq/view.jsp?EID=470744 Julio-2010

[35] Apache TOMCAT
http://tomcat.apache.org/ Septiembre-2010

[36] Apache DERBY
http://db.apache.org/derby/ Septiembre-2010

[37] Servlet
http://www.oracle.com/technetwork/java/index-jsp-135475.html Septiembre-2010

[38] JSP
http://java.sun.com/products/jsp/ Septiembre-2010

[39] NFCIP NFCIP-1. Near Field Communication Interface and Protocol. Estandarizado en el ECMA-340.
http://www.ecma-international.org/publications/standards/Ecma-340.htm Julio-2010

[40] HTTP HTTP 1.1: Hypertext Transfer Protocol. HTTP/1.1.
http://www.w3.org/Protocols/rfc2616/rfc2616.html Septiembre-2010

[41] ISO 14443
http://en.wikipedia.org/wiki/ISO/IEC_14443 Julio-2010